STUDY OF TRAVELING SALESMAN PROBLEM USING GENETIC ALGORITHM

Er. Ashish Gupta[*]

Er. Shipra Khurana[**]

ABSTRACT:

The traveling salesman problem is a permutation problem in which the goal is to find the shortest path between N different cities that the salesman takes is called the TOUR. In other words, the problem deals with finding a route covering all cities so that the total distance traveled is minimal. This paper gives a solution to find an optimum route for traveling salesman problem using Genetic algorithm technique, in which cities are selected randomly as initial population. The new generations are then created repeatedly until the proper path is reached upon reaching the stopping criteria.

Keywords: Genetic Algorithm, TSP, Crossovers, Mutation.

* Asst. Professor, Department Of Computer Science and Technology, RP INDERAPRASTHA INSTITUTE OF TECHNOLOGY, Bastara, Karnal (Haryana), India.

** Lecturer, Department of Computer Science and Technology, RP INDERAPRASTHA INSTITUTE OF TECHNOLOGY, Bastara, Karnal (Haryana), India.

I. INTRODUCTION:

GENETIC ALGORITHM

An Algorithm is a step-by-step procedure to solve a given problem. Genetic algorithms are based on the principle of Genetics and Evolution. “Genetics” is derived from Greek word “genesis” meaning ‘to grow’ or ‘to become’. It is basically a search technique to find approximate solutions to optimization and search problems. Basically, an optimization problem looks really simple. One knows the form of all possible solutions corresponding to particular problem. The set of all possible solutions forms the Search Space. The problem consists of finding out the solution that fits the best. GA handles a population of possible solutions. Each solution is represented through a chromosome, which is just an abstract representation. A set of reproduction operators are applied directly on the chromosomes, and are used to perform mutations and recombination over solutions of the problem.

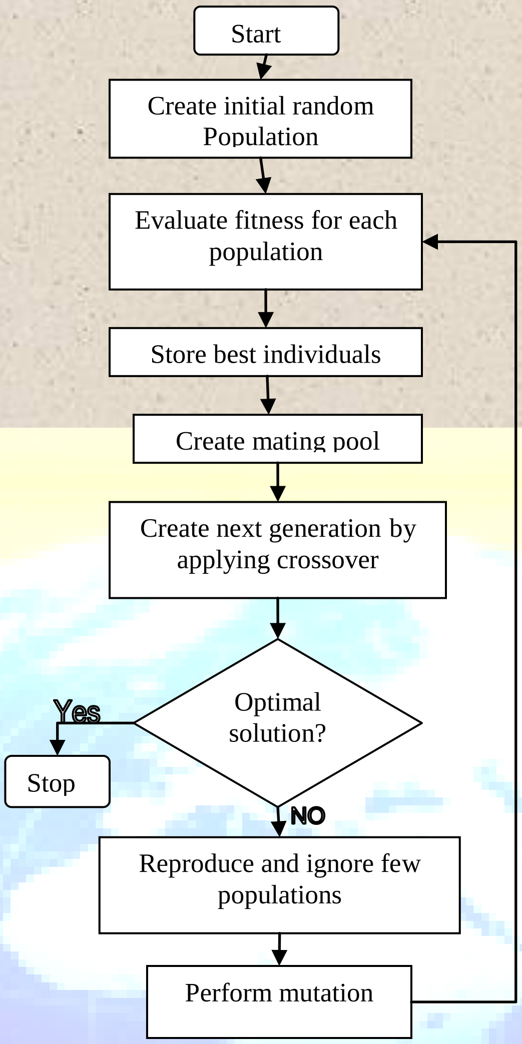

Start
Create initial random Population
Evaluate fitness for each population
Store best individuals
Create mating pool
Create next generation by applying crossover
Optimal solution?
Yes
Stop
NO
Reproduce and ignore few populations
Perform mutation

Figure 1. Genetic Algorithm – Program Flowchart

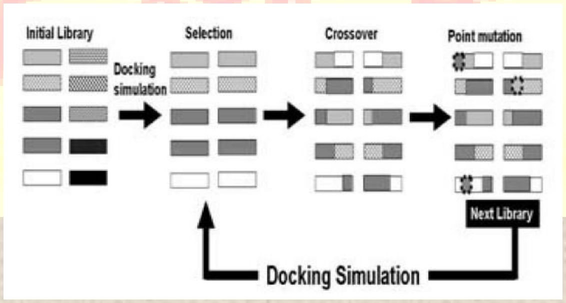


Figure 2. Scheme of Genetic Algorithm

The genetic algorithm loops over an iteration process to make the population evolve. Each iteration consists of the following steps:

1. **Selection**: - The first step consists in selecting individuals for reproduction. The selection is done using various selection techniques (roulette wheel, random, boltzman, etc) based on the relative fitness of the individuals so that best ones are often selected for reproduction.
2. **Reproduction**: - In the second step, selected individuals undergo crossover and mutation for generating offspring's (new chromosomes).
3. **Evaluation**: - In this step the fitness number of the offspring's (new chromosomes) is evaluated.
4. **Replacement**: - During the last step, individuals from the old population are killed and replaced by the new ones.

This algorithm is stopped when the population converges toward the optimal solution.

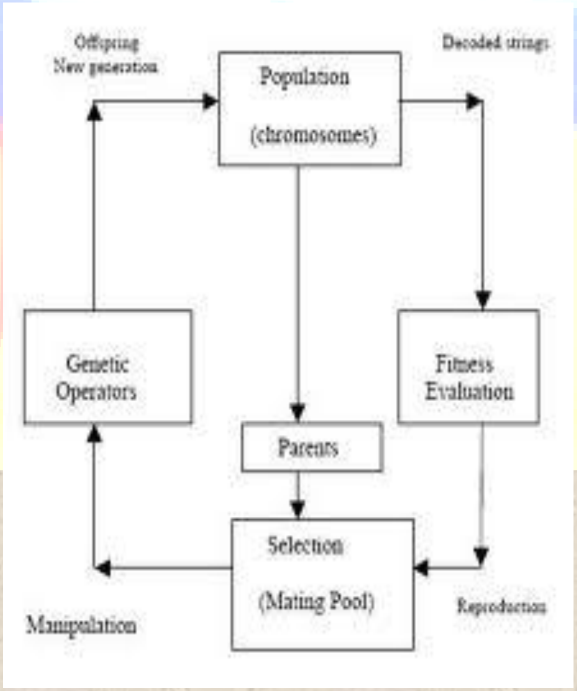


Figure 3. Genetic Algorithm Cycle

II. TRAVELLING SALESMAN:

PROBLEM

It is a permutation problem with the objective of finding the path of the shortest length (or the minimum cost) on an undirected graph that represents cities or node to be visited. The traveling salesman starts at one node, visits all other nodes successively only one time each, and finally returns to the starting node. i.e., given n cities, named {c1, c2, ……., cn}, and permutations, σ1, ..., σn!, the objective is to choose σi such that the sum of all Euclidean distances between each node and its successor is minimized. The successor of the last node in the permutation is the first one. The Euclidean distance d, between any two cities with coordinate (x1, y1) and (x2, y2) is calculated by:

$$d= ((x_1 – x_2)^2+ (y_1-y_2)^2)^{1/2}$$

There are mainly three reasons why TSP has been attracted the attention of many researcher's and remains an active research area. First, a large number of real-world problems can be modeled by TSP. Second, it was proved to be NP-Complete problem. Third, NP-Complete problems are intractable in the sense that no one has found any really efficient way of solving them for large problem size. Also, NP-complete problems are known to be more or less equivalent to each other; if one knew how to solve one of them one could solve the lot. The methods that provide the exact optimal solution to the problem are called exact methods. An implicit way of solving the TSP is simply to list all the feasible solutions, evaluate their objective function values and pick out the best. However it is obvious that this "exhaustive search" is grossly inefficient and impracticable because of vast number of possible solutions to the TSP even for problem of moderate size. Since practical applications require solving larger problems, hence emphasis has shifted from the aim of finding exactly optimal solutions to TSP, to the aim of getting, heuristically, 'good solutions' in reasonable time and 'establishing the degree of goodness'. Genetic algorithm (GA) is one of the best heuristic algorithms that have been used widely to solve the TSP instances.

a) Genetic coding

To apply GA for any optimization problem, one has to think a way for encoding solutions as feasible chromosomes so that the crossovers of feasible chromosomes result in feasible chromosomes. The techniques for encoding solutions vary by problem and, involve a certain amount of art. For the TSP, solution is typically represented by chromosome of length as the number of nodes in the problem.

Each gene of a chromosome takes a label of node such that no node can appear twice in the same chromosome. There are mainly two representation methods for representing tour of the TSP – array representation and path representation. We consider the path representation for a tour, which simply lists the label of nodes. For example, let {1, 2, 3, 4, 5} be the labels of nodes in a 5 node instance, then a tour {1→ 3→ 4→ 2→ 5 → 1} may be represented as (1, 3, 4, 2, 5). All the cities are sequentially numbered starting from one. The route between the cities is described with an array with each element of the array representing the number of the city. The array represents the sequence in which the cities are traversed to make up a tour. Each chromosome must contain each and every city exactly once.

1	4	2	6	5	3

This chromosome represents the tour starting from city 1 to city 4 and so on and then back to city 1.

b) Fitness Function

A fitness function evaluation is incorporated to assigns a value to each organism, noted as fi. This fi value is a figure of merit which is calculated by using any domain knowledge that applies. In principle, this is the only point in the algorithm that domain knowledge is necessary. Organisms are chosen using the fitness value as a guide, where those with higher fitness values are chosen more often. Selecting organisms based on fitness value is a major factor in the strength of GAs as search algorithms. The method employed here was to calculate the total Euclidean distance Di for each organism first, then compute fi by using the following equation

$$fi = Dmax - D_i$$

where Dmax is the longest Euclidean distance over organisms in the population.

c) Selection

The selection operator chooses two members of the present generation to participate in the next operations of crossover and mutation. There are two popular approaches for implementing selection, the first called "roulette selection" and the second is "random selection".

Roulette wheel selection: - It is also known as Fitness- Proportionate Selection, is a genetic operator, used for selecting potentially useful solutions for recombination. In this, the chance of an individual's being selected is proportional to its fitness, greater or less than its competitor's fitness. Conceptually it can be thought of as a game of roulette.

$$pi = Fi \,/ \sum_{j=1}^{n} Fj$$

Random selection: - This technique randomly selects a parent from the population. In terms of disruption of genetic codes, random selection is a little more disruptive, on average, than roulette wheel selection.

The selection operator then assures that each organism participates as parent exactly Si times. The system performance using the above two methods was poor; in the sense that it takes long time to diverge. This result implied to adopt the following newly developed method; which is a combination of the above two methods with some modification, so that relevant organisms with higher fitness are selected and survived.

Selection raises the issue of fitness function for TSP. A requirement of the above selection methods is that fi be higher for organisms that represent better tours. However tours physically comprise lower Euclidean distances.

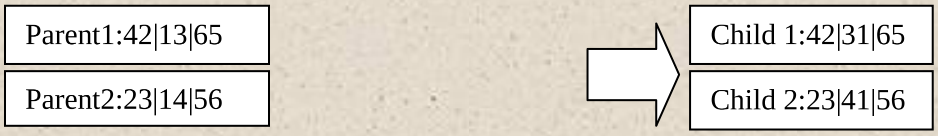
d) Crossover

To solve the traveling salesman problem, a simple crossover reproduction scheme does not work as it makes the chromosomes inconsistent i.e some cities may be repeated while others are missed out. To avoid these three types of special crossover operators reported for permutation problems are selected to be examined and used in the proposed TSP system. They are: order crossover (OX), partially matched crossover (PMX), and cycle crossover (CX), as described briefly below.

Order Crossover (OX)

To apply OX, two random cross points are selected. Alleles from parent1 that fall between the two cross points are copied into the same positions of the offspring. The remaining allele order is determined by parent2. Non duplicative alleles are copied from parent2 to the offspring beginning at the position following the second cross point. Both the parent2 and the offspring are traversed circularly from that point. A copy of the parent′s next non duplicative allele is placed in the next available child position. Ordered two-point crossover is used when the problem is of order based, for example in U-shaped assembly line balancing etc. given two parent chromosomes, two random crossover points are selected partitioning them into a left, middle and right portion. The ordered two-point crossover behaves in a following way: Child 1 inherits its left and right section from parent 1, and its middle section is determined by the genes in the middle section of parent 1 in the order in which the values appear in parent 2. A similar process is applied to child 2. This is shown in example below.

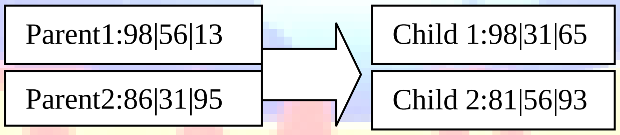
Example



Partially Matched Crossover

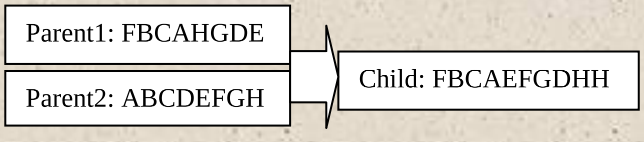
PMX proceeds just as OX. Alleles from parent1 that fall between two randomly selected crossing sites are copied into the same positions of the offspring. The remaining allele's positions are determined by parent2 during a two step process. First, alleles in parent2 not within crossing sites are copied to the corresponding positions within the offspring. Next each allele of parent2 within the crossing sites is placed in the offspring at the position occupied in parent2 by the allele from parent1 that displaced it, see the example below. PMX can be applied usefully in the TSP. Indeed, TSP chromosomes are simply sequences of integers, where each integer represents a different city and the other represents the time at which the city is visited. Under this representation, known as permutation encoding, we are only interested in labels and not alleles. It may be viewed as a crossover of permutation that guarantees that all positions are found exactly once in each offspring.

Example



Cycle Crossover

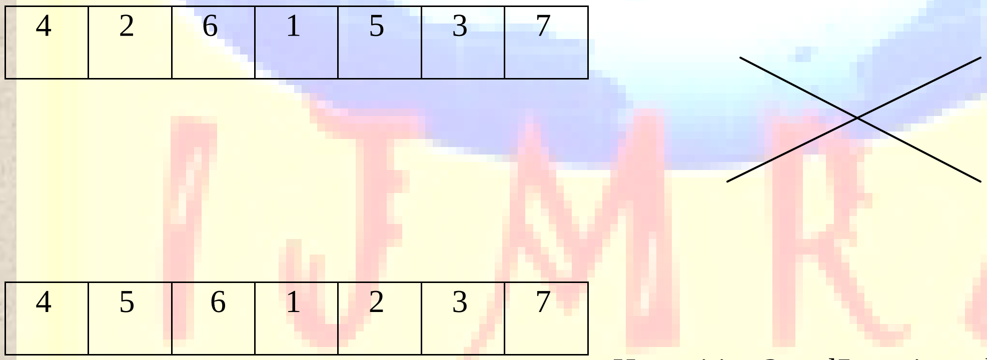
CX performs recombination under the constraint that each allele value comes from one parent or the other. CX does not use crossing sites, but a cycle is defined in a manner similar to an algebraic permutation group, see the example below, Comparing the strings of parent1 with parent2, F displaces A, A displaces D, D displaces G, and displaces F. This forms the cycle FADG. The remaining alleles are filled from parent2 in the corresponding positions, BC, E, and H.



Example

e) Mutation

The mutation operator randomly selects a position in the chromosome and changes the corresponding allele, thereby modifying information. The need for mutation comes from the fact that as the less fit members of successive generations are discarded; some aspects of genetic material could be lost forever. By performing occasional random changes in the chromosomes, GAs ensure that new parts of the search space are reached, which reproduction and crossover alone couldn't fully guarantee. In doing so, mutation ensures that no important features are prematurely lost, thus maintaining the mating pool diversity. Mutation has a high probability of resulting in a non-viable city order. However, mutation is still applied by accounting for the non-viable city orders in the evaluation function. For this problem, mutation refers to a randomized exchange of cities in the chromosomes. For instance, example shown in fig.



Here cities 2 and5 are interchanged because of an inversion operation.

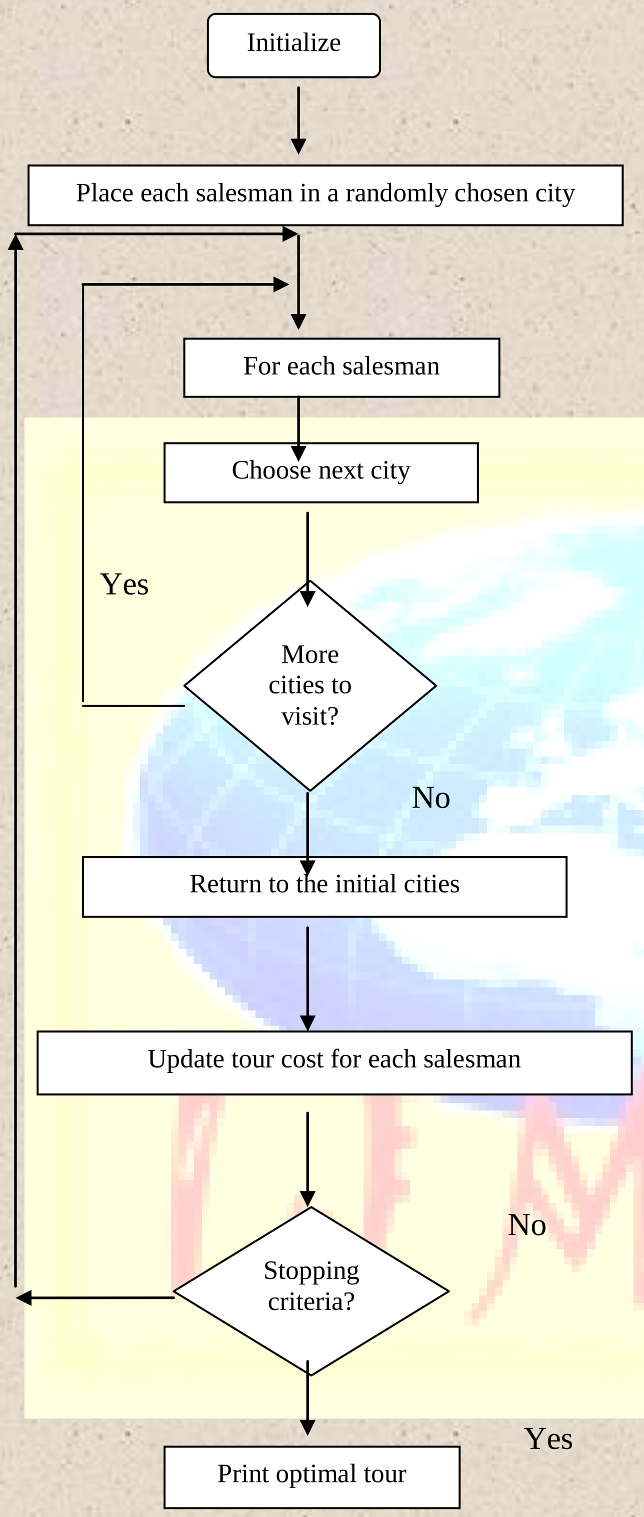
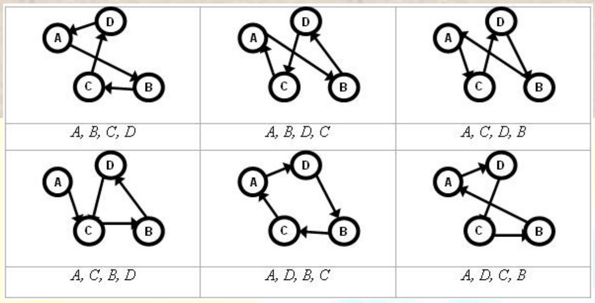


Figure 4. Flow Chart for Travelling Salesman Problem

Considering four cities as A, B, C and D. A salesman needs to visit all these cities and return home to city A. In which order should he visit the cities in such a way that the total distance is minimized? All possible configurations have considered find the optimal solution. For our example the possible solutions are given below.



As can be seen from the figure for N cities there are (N -1)! possible configurations. When N is small like the one in the example, it is very easy to find the optimal solution. However, when N becomes large the need for a computer becomes obvious. It could even be hard for a fast computer to obtain the result when N is large enough.

III. CONTROL PARAMETERS:

These are the parameters that govern the GA search process. Some of them are:

(a) Population Size: - It determines how many chromosomes and thereafter, how much genetic material is available for use during the search. If there is too little, the search has no chance to adequately cover the space. If there is too much, the GA wastes time evaluating chromosomes.

(b) Crossover probability: - It specifies the probability of crossover occurring between two chromosomes.

(c) Mutation probability: - It specifies the probability of doing bit-wise mutation.

(d) Termination criteria: - It specifies when to terminate the genetic search.

IV. CONCLUSION:

In this paper we have discussed the travelling salesman problem using Genetic Algorithm. Various techniques of genetic algorithm have been discussed in this paper to study travelling salesman problem which is a permutation problem in which goal is to find the shortest path between cities traversing each city at least once. This paper gives a solution to find an optimum route for traveling salesman problem using Genetic algorithm technique, in which cities are selected randomly as initial population. The new generations are then created repeatedly until the proper path is reached upon reaching the stopping criteria.

V. REFERENCES:

- C.H. Papadimitriou and K. Steglitz. “Combinatorial Optimization: Algorithms and Complexity”. Prentice Hall of India Private Limited, India, 1997.
- C.P. Ravikumar. "Solving Large-scale Travelling Salesperson Problems on Parallel Machines”. Microprocessors and Microsystems 16(3), pp. 149-158, 1992.
- R.G. Bland and D.F. Shallcross. "Large Travelling Salesman Problems arising form Experiments in X-ray Crystallography: A Preliminary Report on Computation". Operations Research Letters 8, pp. 125-128, 1989.
- D.E. Goldberg and R. Lingle. “Alleles, Loci and the Travelling Salesman Problem”. In J.J.Grefenstette (ed.) Proceedings of the 1st International Conference on Genetic Algorithms and Their Applications. Lawrence Erlbaum Associates, Hilladale, NJ, 1985.
- L. Davis. “Job-shop Scheduling with Genetic Algorithms”. Proceedings of an International Conference on Genetic Algorithms and Their Applications, pp. 136-140, 1985. Zakir H. Ahmed International Journal of Biometrics & Bioinformatics (IJBB) Volume (3): Issue (6) 105
- I.M. Oliver, D. J. Smith and J.R.C. Holland. “A Study of Permutation Crossover Operators on the Travelling Salesman Problem”. In J.J. Grefenstette (ed.). Genetic Algorithms and Their Applications: Proceedings of the 2nd International Conference on Genetic Algorithms. Lawrence Erlbaum Associates, Hilladale, NJ, 1987.

- D. Whitley, T. Starkweather and D. Shaner. "The Traveling Salesman and Sequence Scheduling: Quality Solutions using Genetic Edge Recombination". In L. Davis (Ed.) Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York, pp. 350-372, 1991.
- N.J. Radcliffe and P.D. Surry. "Formae and variance of fitness". In D. Whitley and M. Vose (Eds.) Foundations of Genetic Algorithms 3. Morgan Kaufmann, San Mateo, CA, pp. 51-72, 1995.
- P. Poon and J. Carter. "Genetic algorithm crossover operations for ordering applications". Computers and Operations Research 22, pp. 135–47, 1995.
- I. Choi, S. Kim and H. Kim. "A genetic algorithm with a mixed region search for the asymmetric traveling salesman problem". Computers & Operations Research 30, pp. 773 – 786, 2003.
- C. Moon, J. Kim, G. Choi and Y. Seo. "An efficient genetic algorithm for the traveling salesman problem with precedence constraints". European Journal of Operational Research 140, pp. 606-617, 2002.
- D.E. Goldberg. "Genetic Algorithms in Search, Optimization, and Machine Learning". Addison- Wesley, New York, 1989.
- K. Deb. "Optimization For Engineering Design: Algorithms And Examples". Prentice Hall Of India Pvt. Ltd., New Delhi, India, 1995.
- Z.H. Ahmed. "A sequential Constructive Sampling and Related approaches to Combinatorial Optimization". PhD Thesis, Tezpur University, India, 2000.
- Z.H. Ahmed and S.N.N. Pandit. "The travelling salesman problem with precedence constraints". Opsearch 38, pp. 299-318, 2001.
- TSPLIB, http://www.iwr.uni-heidelberg.de/iwr/comopt/software/TSPLIB95/