Minimization of travel cost of 4D TSP using Genetic Algorithm and a comparison study of selection, crossover and Mutation methods for TSPLIB instances.

Midnapore college (Autonomous)
Depertment of Computer Science
Guide : Assistant Professor Shovan Roy (Department of CSA)
Group : Arun Sarma — Rohan Das — Suman Paik — Sumit Mandal

July, 2022

Abstract The Traveling Salesman Problem is a classic combinatorial optimization problem with a straightforward notion but a challenging solution. Suppose we have n number of cities, each city is connected with other cities, here the condition is one who wants to travel through all the cities only for one time and back to the starting city, here we will calculate the minimal cost of travel. The possible routes is n!(n=city count). If we have multiple routes between each cities then the problem will become a 3D TSP, and if we have multiple routes and multiple vehicles to ride, then it will become a 4D TSP. That is why TSP is called a NP-hard problem. Solving TSP through brute force is very time inefficient. To solve problems like this, we can use Genetic Algorithm. This algorithm can be used in various kind of optimization problems, TSP is one of them. Here in this paper we will be discussing Symmetric TSP, Genetic Algorithm and it's procedure to solve TSP, 4D TSP, different algorithms used in Genetic Algorithm.

Keywords Travelling salesman problem,2D,3D,4D model, symmetric TSP, Genetic algorithm, Crossover Mutation Operator,Fitness value Shortest root, Selection, Evolution.

1 Introduction

In this TSP, Suppose we have n number of cities. Each city visited exactly once and finally return to the starting city. Here we try to find the minimum travel distance between those cities. In the symmetric travelling salesman problem, there is a difference in distance between two cities when travelling in the opposite direction, the costs may vary, at times only one-way traffic is allowed on the way from one city to another. Although the assertion is easy to say but it is more challenging to resolve. Travelling Salesman Problem is an optimization problem and there is a huge search space. It is called NP-hard which means in polynomial time, it cannot be resolved. TSP is being applied in many cases such as microchip making, vehicle routing, drilling on printed circuit boards, Overhauling gas turbine engines etc. let's say we have n numbers of cities, and then we can find (n - 1)! different routes to cover all the cities. With the help of TSP, we can find the shortest path.

2 Literature review

The origins of the travelling salesman problem are obscure. The topic was mentioned in an 1832 handbook for travelling salesmen problem, which gives example tours through Germany and Switzerland, but there is no mathematical treatment. The travelling salesman problem was expressed

mathematically by W.R. Hamilton(Irish mathematician) and Thomas Kirkman(British mathematician) in the 19th century. TSP has a basic concept, however it is more complex to solve.[11] TSP is an optimization problem with a large search space that is NP-hard, meaning it cannot be solved in polynomial time. It is currently one of the most essential issues in the field of computer science. Several heuristic methods, including the greedy approach, ant algorithms, simulated annealing, tabu search, and evolutionary algorithms, have been used to identify the best solution to these problems. [1]. When the no of cities increases, computing the solution gets more complicated. Despite the computing challenges, technologies such as evolutionary algorithms called genetic algorithms that can provide a near-optimal solution for thousands or more cities. In this paper, we give an overview of different genetic operators like selection, crossover, mutation and other stuff to solving the travelling salesman problem. By combining crossover and mutation operators, we are also able to express the experimental results in relation to path representation with a variety of standard examples.

The majority of what we now refer to as genetic algorithms were created by John Holland, even though computer models of evolutionary processes date back to the 1950s (also known as "GAs"). He is a professor at the University of Michigan, whose book Adaptation in Natural and Artificial Systems pioneered GA research. Nowadays the genetic algorithm is a part of a wider field of research, often referred to as "Evolutionary Computing." In computer science and operations research, a Genetic Algorithm (GA) is a metaheuristic that belongs to the larger class of evolutionary algorithms and is inspired by natural selection (EA). To develop high-quality solutions to optimization and search problems, genetic algorithms use biologically inspired operators such as mutation, crossover, and selection. Researchers are continuously trying to improve the Genetic Algorithm methods or the single operators for improving the overall performance of this algorithm.

The researchers are continuously trying to find a better way to solve the TSP problem with the help of Genetic Algorithm. There is an exciting review paper about the selection method of Genetic Algorithm which was used to get the result of Land Suitability [2]. In another case other researchers had worked on the topic of genetic algorithm which contain the different selection methods [3]. Researchers were trying to solve this TSP using different selection, crossover[12], mutation operators [4] [5]. There is an excellent book which can build the concept of genetic algorithm in a better way called "Introduction of Genetic Algorithm" [6]. In recent years TSP solutions have been attempted using genetic algorithms by many researchers. The biological terminology is applied to this algorithm. This means that both nature and genetic algorithms work on populations of individuals that are defined by one or more mathematical genes on the chromosomes.

The genetic algorithm (GA) follows these steps:

1. Generate a population: The GA generates a set of chromosomes by randomly sampling values of changing cells between the bottom and upper boundaries. The population is the original set of chromosomes.

2. Create a new generation: Chromosomes with a lower fitness function have a higher probability of surviving to the next generation in the new generation. To create chromosomes for the next generation, crossover and mutation are used.

3. Stopping conditions: The algorithm repeats step-2 for each generation, recording the best value of the fitness function in that generation. The GA terminates when there is no improvement in the best fitness value after a number of generations. [7]

Operators	Author's
Tournament + Proportional Roulette Wheel Selection	Noraini Mohd Razali, John Geraghty
Rank-based selection	S.N.Sivanandam, S.N.Deep
Order Crossover	I. Korejo , S. Yang and C. Li
Cyclic Crossover	S.N.Sivanandam, S.N.Deep
Partially Mapped Crossover	Gold- berg and Lingle (1985).
Multi-Parent Crossover	S.N.Deep
Inversion Mutation	Fogel (1990)
Insertion Mutation	Fogel (1988)
Exchange Mutation	W. Banzhaf (1990)

3 Travelling Salesman Problem

The Travelling Salesman Problem is a NP-hard problem[10], meaning that no exact algorithm exists to solve this kind of problem in polynomial time. The predicted time to find the best solution is exponential.The shortest path problem is characterised as a combinatorial problem with the goal of finding the shortest path (or the minimum cost). Cities are the graph's vertices, pathways are the graph's edges, and a path's distance is the edge's length, hence TSP can be represented as an unsupervised weighted graph. It's a minimization issue that starts and ends at a certain vertex after only visiting each other vertex once. [5]

Variable name	Description
V	set of nodes
E	set of edges
N	no of cities
i,j	name of cities
C_{ij}	the cost of travel from the i-th city to j-th city
t_{ij}	the time for travelling from the i-th city to j-th city
v	vehicle
r	route
k	type of conveyance
P_i	selection probability for individual i
R_i	rank of i-th individuals
G	maximum generation
g	current generation number

3.1 Classical TSP Model

TSP can be written as graph G = (V, E) in a classical two-dimensional TSP, where A = 1, 2,..., N is the set of nodes, and E is the set of edges. A salesman has to travel to N cities at minimum cost. In this tour, the salesman begins in one location, visits each city exactly once, and then returns to the beginning city at minimum cost. Let C_{ij} represent the cost of travel from the i-th city to j-th city. Then the model is mathematically formulated as

$$\text{Determine a complete tour} \quad x_{ij}, i = (1, 2, ..., N), j = (1, 2, ..., N)$$

$$\text{minimize} \quad Z = \sum_{i=1}^{N-1} \sum_{j=1}^{N-1} x_{ij} c_{ij}$$

3.2 3D TSP Model

Let c(i, j, r) represent the cost of travelling by the r-th route from city i-th to city j-th. The salesman then must decided for a complete tour (x_1, x_2,..., x_N , x_1) and corresponding available route types

(r_1, r_2, ..., r_s) for the tour, where $x_i \in \{1, 2, ...N\}$ for i = 1,2,...,N, $r_i \in \{1, 2, ..s\}$ for i = 1,2,...,N and all x_i's are distinct. The problem can be expressed mathematically as:

$$\text{minimize} \quad Z = \sum_{i=1}^{N-1} c(x_i, x_{i+1}, r_i) + c(x_N, x_1, r_1),$$

$$\text{where} \quad x_i \neq x_j, i, j = 1, 2, ...N, \qquad r_i, r_1 \in 1, 2, ..ors)$$

3.3 4D TSP Model

Let c(i, j, r, v) represent the cost of travelling the v-th type of conveyance on the r-th route from city i-th to city j-th. The salesman then must decided for a complete tour (x_1, x_2,..., x_N, x_1) and the corresponding available route types (r_1, r_2, ..., r_s) and conveyance types (v_1, v_2, ..., v_p) to be used for the tour, where $x_i \in$ 1,2, ..N for i = 1, 2, ..., N, $r_i \in \{1, 2, ..s\}$ and $v_i \in \{1, 2, ..p\}$ for i = 1,2, ..., N and all x_i's are distinct. Then the problem can be expressed mathematically as:

$$\text{minimize} \quad Z = \sum_{i=1}^{N-1} c(x_i, x_{i+1}, r_i, v_i) + c(x_N, x_1, r_1, v_1),$$

$$\text{where} \quad x_i \neq x_j, i, j = 1, 2, ...N, \qquad r_i, r_1 \in 1, 2, ..ors), \quad v_i, v_1 \in (1, 2, ..orp)$$

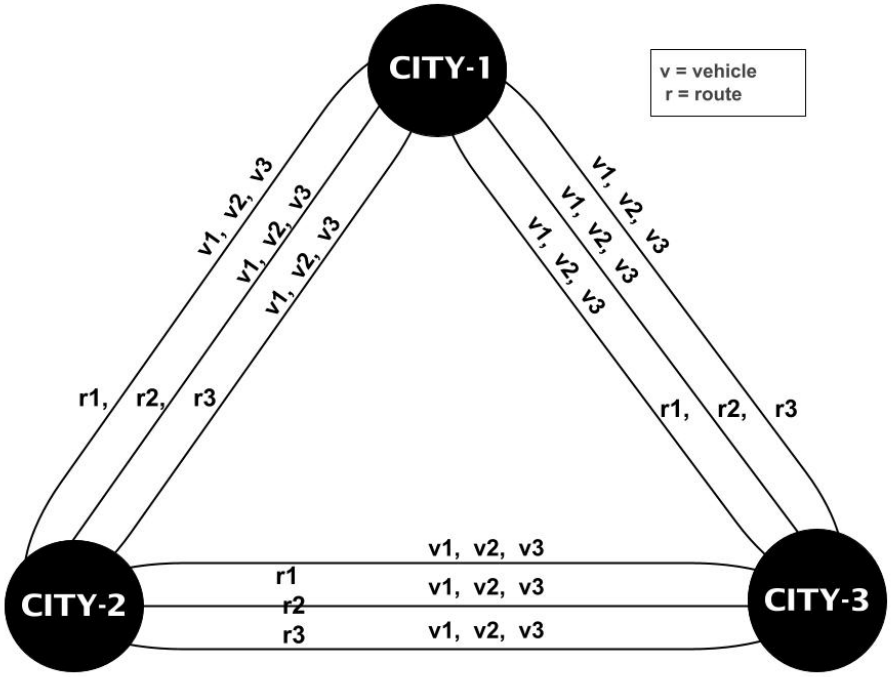


Figure 1: 4D TSP Model

4 Genetic Algorithm

A genetic algorithm is a heuristic search algorithm used to solve search and optimization problems. This algorithm can solve problems very fast which if you try to solve by brute force that would take too much time for it. This algorithm can be used in many real-life applications such as data centres, image processing, code-breaking, electronic circuit design, and artificial creativity. Genetic algorithms are inspired by the concept of genetics and natural selection to provide best solutions to problems. GA is based on the behaviour of chromosomes and their genetic structure. Here each chromosome

participates with a vital role to produce the possible solutions. Here the fitness function is used to provide the characteristics of all individuals in the whole population. When the function is bigger then the solution is much more accurate. That's why these algorithms have better intelligence than random search algorithms because these algorithms use historical data to take the search to the best performing region within the solution space [6].

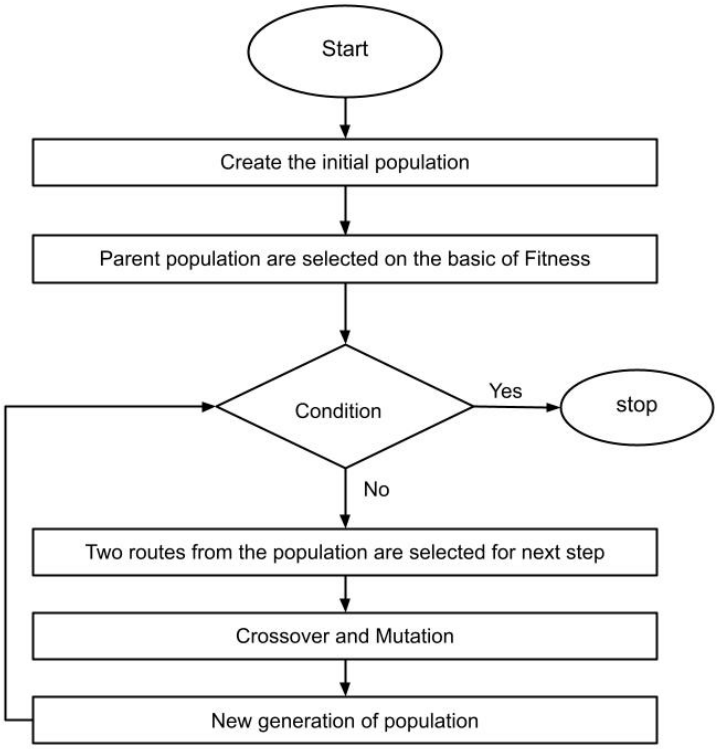


Figure 2: Genetic Algorithm Flowchart

Algorithm 1 Fundamental Genetic Algorithm [9]

Genetic Algorithm()
{
Create a Initial population using a function, called **Random( )**
Calculate the fitness value of each individuals using a function, named **Fitness( )**
While the fitness criteria is not satisfied
do
{
Selection of two routes from the population for next reproduction using the select function()
Selection Parent-one, Parent-two

Crossover Perform the crossover among them(parent-one and parent-two).

Perform the mutation operation on the newly generated child with the help of mutation.

Mutation(M child route

Calculate the **fitness value** of child chromosome and replace the parent population with it.
}
}

5 Genetic Algorithm follows the following phases to solve complex optimization problems

To begin, initial population is first created in genetic algorithm. This initial population contains all possible solutions to the given problem. Most common technique, random binary strings used for initialization.

5.1 Fitness assignment

The fitness function help to determine the population's overall fitness. Every individual is assigned a fitness score, which determines the chances of those individuals being selected for reproduction. If someone's fitness score is better then his chances of being selected for reproduction are high.

5.2 Selection Strategy for Reproduction

The chosen individuals are grouped in pairs of two to enhance reproduction. The genes of these people are passed down to the following generation. Identifying the region with the best possible chance of coming up with the best solution to the problem is the main objective of this step (better than the previous generation). The genetic algorithm makes use of the fitness proportionate selection approach to make sure that effective recombination solutions are used. The selection operator ensures the better members of the population (with higher fitness) so that the better members can reach the next generation. So there are various kinds of selection strategies. Some popular methods are given below.

5.2.1 Tournament Selection

Tournament Selection is one of the most popular selection methods in genetic algorithms. In this method the chance of selecting individuals are performed based on their fitness values. The primary idea behind this technique is to select the individual with the highest fitness value from a particular number of chromosomes in the population into the next generation. There is no arithmetical computation based on fitness value in the tournament selection, only a comparison between individuals based on fitness value. Here Tournament size refers to the number of chromosomes that will be competing in the tournament. This technique is faster than the roulette wheel and Rank Based selection method [3].

Algorithm 2 Tournament Selection

Step 1: We select the tournament size n at random.

Step 2: We randomly select n people from the population, and we then decide which one has the highest fitness values.

Step 3: The best chromosome is transferred in the mating pool.

Step 4: Consequently, in this plan, each tournament only selects one chromosome and N_p tournaments are to be played to make the size of the mating pool equal to N_p

Input Consider Population size N with fitness values. Output: Mating pool size N_p ($N_p \leq$ N). Steps: 1) Select N_u individuals at random ($N_u \leq$ N). 2)Out of N_u individuals, choose the individual with the highest fitness value as the winner. 3) Add the winner to the mating pool, which is initially empty. 4) Repeat Steps 1-3 until the mating pool contains N_p, individuals 5) Stop Example of Tournament Selection method : Here, N = 8, $N_u = 2$, $N_p = 8$

Individual	1	2	3	4	5	6	7	8
Fitness	**2.0**	**3.1**	**4.1**	**5.0**	**5.6**	**2.9**	**2.8**	**5.5**

Output

N_u trial individual	Individual pair	Selected Individual
1	2,4	4
2	3,8	8
3	1,3	3
4	4,5	5
5	1,6	6
6	1,2	2
7	4,2	4
8	8,3	8

5.2.2 Proportional Roulette Wheel Selection

Individuals are selected using the proportional roulette wheel selection method with a probability that is directly proportional to their fitness values, meaning that their selection corresponds to a specific portion of the roulette wheel. The probability of selecting a parent can be viewed as spinning a roulette wheel, with the size of the segment for each parent being proportional to it's fitness. Individuals with the largest fitness value have the higher probability of being selected. Within the roulette wheel, the fittest individual occupies the largest segment, while the least fit occupies a smaller segment. The sum of all the individuals' fitness values makes up the roulette wheel's circumference. When the wheel is spun, it will eventually come to a halt, and the pointer attached to it will point at one of the segments, most likely the broadest. All segments, on the other hand, have a chance, with a probability proportionate to their width. By repeating this process each time an individual must be picked, the better individuals will be chosen more frequently than the inferior ones, ensuring that the survival of the fittest conditions are met. Let's consider f1, f2,. . . , f n are the fitness values of the individuals 1, 2,. . . , n. Selection probability (P_i) for individual i is -

$$P_i = \frac{f_i}{\sum_{j=1}^{n} f_j}$$

The primary benefit of proportional roulette wheel selection is that it does not dismiss any individuals from the population and allows all of them to be chosen. As a result, population diversity is preserved. However, there are a few key flaws in proportional roulette wheel selection. Outstanding individuals will represent bias at the start of the search, which could lead to an early convergence and loss of diversity. If an early population includes one or two very fit but not the best individuals, and the remainder of the population is poor, these fit individuals will quickly dominate the population, preventing the population from exploring other potentially better individuals.Such a great domination results in a significant loss of genetic variety, which is unfavourable to the optimization process.

Algorithm 3 Proportional RW-Selection [3]

```
while population size < popsize do
    Generate popsize random number r
    Calculate cumulative fitness, total fitness ( P_i ) and sum of proportional fitness (Sum)
    Spin the wheel popsize times
    if Sum < r then
        Select the first chromosome, otherwise. select j-th chromosome
    end if
end while
Return chromosomes with fitness value proportional to the size of selected wheel section
End Procedure
```

5.2.3 Rank-based selection

Rank-based selection is a selection method in which a chromosome's probability of being chosen is determined by using the fitness ranking in comparison to the overall population. In this selection method at first seperate and choose the individuals from the population based on their fitness before computing selection probabilities based on their ranking value rather than fitness values whereas the proportional Roulette wheel selection considers the fitness value and computes probability based on

it and the best individual heavily dominates the entire population, resulting in a significant loss of genetic variety,which is not beneficial to the process of optimization.This is one of the biggest problems of the roulette wheel selection.To overcome this particular problem here we will use the rank-based selection. The process of this rank selection defines through these two steps. 1. The fitness values of the individuals are displayed in increasing order. The chromosome contains with the lowest fitness value is assigned by rank 1, and the others are assigned through their fitness values. 2. The proportionate based selection scheme is then followed based on the assigned rank [6]. The area to be occupied by a particular individual i, is given by-

$$\frac{R_i}{\sum_{i=1}^{n} R_i}$$

Where R_i indicates the rank of the i - individual. The rank of two or more people with the same fitness values should be the same. Rank-based Selection Example : There are 4 individuals and their fitness values are given below f1 = 0.40, f2 = 0.05, f3 = 0.03, f4 = 0.02. Their proportionate areas on the wheel are : 80 %, 10 %, 6 %, and 4 %

Individual (i)	Fitness ()	RW (Area)	Rank	RS (Area)
1	0.60	80%	4	40%
2	0.07	10%	3	30%
3	0.05	6%	2	20%
4	0.02	4%	1	10%

Algorithm 4 Rank-based Selection

while population size < popsize **do**
  Sort population according to rank
  Assign fitnesses to the individuals according to linear rank function
  Generate popsize random number (r)
  Calculate cumulative fitness,total fitness and sum of proportional fitness (Sum)
  Spin the wheel popsize times
  if Sum < r **then**
    Select the first chromosome, otherwise. select j-th chromosome
  end if
end while
Return chromosomes with fitness value proportional to the size of selected wheel section
End Procedure

This is how the rank-based selection method overcomes the drawback of proportional roulette wheel selection without loss of genetic diversity.

5.2.4 Stochastic Universal Sampling (SUS Selection)

This particular selection method is similar to the RW selection method, where all individuals occupy spaces according to their fitness value. But instead of selecting just one individual at a time here the selection will be multiple at a time in fact the desired number selection can be done at once from the population. In RW selection there is one fixed point to select the individual and in this selection the fixed point can be more than one. Therefore all the individuals are chosen in just one single spin of the wheel. And performing this operation encourages the highly fit individuals to be chosen at least once.

Algorithm 5 Stochastic Universal Sampling (SUS Selection)

```
SUS(Population, N)
    F := total fitness of Population
    N := number of offspring to keep
    P := distance between the pointers (F/N)
    Start := random number between 0 and P
    Pointers := [Start + i*P — i in [0..(N-1)]]
    return RWS(Population,Pointers)
RWS(Population, Points)
    Keep = []
    for P in Points
        i := 0
        while fitness sum of Population[0..i] ¡ P
            i++
        add Population[i] to Keep
    return Keep
```

5.3 Crossover

The reproductive and biological crossover operations [12] are analogous to the crossover operator. In this, more than one parent is chosen, and using the genetic makeup of the parents, one or more offspring are created. In a GA, crossover is typically used with a high likelihood. There are various kind of crossover like -

5.3.1 Order Crossover

In this situation of order crossover, a piece of the first parent's chromosome gets copied to the offspring's chromosome, and the offspring inherits the remaining values. Now select two cut points randomly in each parent's genes.The consecutive alleles between two cut points from Parent 1 are copied to the Offspring 1. Then, the remaining genes are copied from the second cut point in the order in which Parent 2 starts after the second cut point till all positions are filled. The concept is to keep the pieces in their relative sequence. The internal concept is mentioned bellow-

Algorithm 6 Order Crossover (OX) [3]

Step 1: Pick a random section from the first parent.

Step 2: Give the first child a copy of this section.

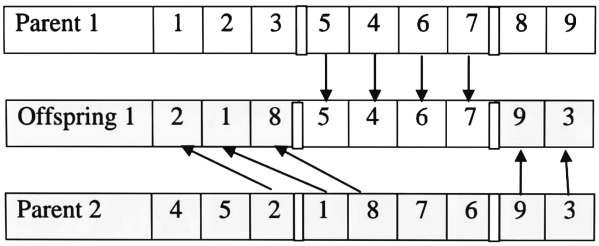
Step 3: To the first child,copy the numbers that aren't in the first part.

(i) beginning at the duplicated part's cut point,

(ii) utilising the second parent's sequence,

(iii) and wrapping around at the end

Step 4: Reverse the parent roles for the second child



5.3.2 Cyclic Crossover

The cyclic crossover procedure is used for simple TSP. It is focused on subsets of cities that take possession of both parents in the same subset of positions. These cities are then copied from the first parent to the child, and the remaining positions are filled with the cities of the second parent. As a result, each city's position is inherited from one of the two parents. The initial subset of cities is not always positioned at consecutive points in the parent tours, therefore numerous edges can be broken during the process.

Algorithm 7 Cyclic Crossover (CX) [6]

Step 1: Identify the gene in the first position of the second parent and go to the corresponding gene in the first parent.

Step 2: Move vertically from the current gene of the first parent to the gene of the second parent.

Step 3: Check if the second parent's gene is the same as the first parent's first gene. If yes, then go through step-5; If not, then go through step-4.

Step 4: Go to the first parent gene that corresponds to the current gene of the second parent and go to step-2.

Step 5: Repeat the same steps to get the second offspring .

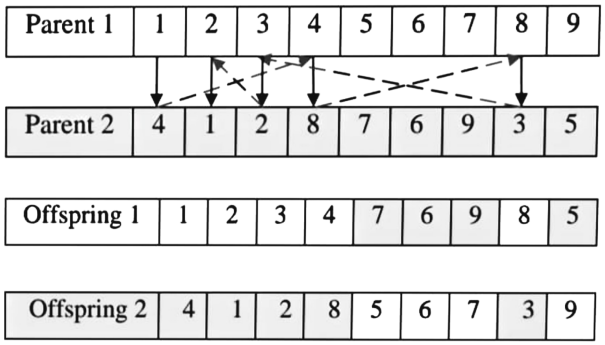
Step 6: Copy the genes present in the first cycle to the same position as the first offspring.

Step 7: Copy the genes present in the second parent cycle to the same position as the second offspring.

Step 8: Copy the remaining genes of the second parent to the same position of the first offspring as shown.

Step 9: Copy the remaining genes of the first parent to the same position as the second offspring.

Step 10: The current sequence of genes in each offspring forms the final similar offspring.



5.3.3 Partially Mapped Crossover (PMX)

PMX stands for partially matched crossover, and it works in a similar way to two-point crossover. Two crossover points are chosen among two parents in this procedure, and the genes between the two crossover points are exchanged, while the remaining genes are filled by partial mapping.[8]

Algorithm 8 Partially Mapped Crossover (PMX)

Step 1: Copy a random portion from P1 by choosing it(P1 and P2 parents).

Step 2: Look for elements in that segment of P2 that have not been copied, starting with the first crossing point.

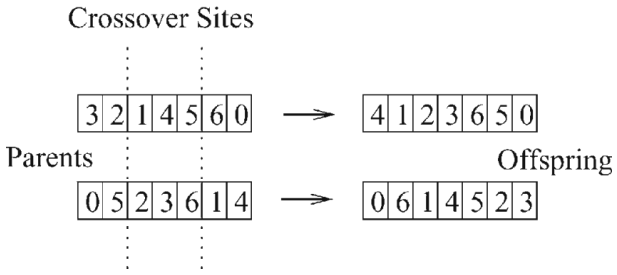
Step 3: For each of these i look in the offspring to see what element j has been copied from P1 in its place.

Step 4: Put i in P2 location that j previously held. since we know that we won't be putting j there (as is already in offspring).

Step 5: Put i in the position held by k in P2 if the space occupied by j in P2 has already been filled by the offspring k.

Step 6: Having dealt with the elements from the crossover segment, the rest of the offspring can be filled from P2.

Step 7: Second child is created analogously.



5.3.4 Multi-Parent Crossover

Adaptation is a fairly popular topic in our culture nowadays as a result of various practical situations. Aside from the original parents (father and mother), there are two additional parents (father and mother) who have been included. In order to generate offspring, a new strategy with 4 parents (the first 2 are original parents and next 2 are adoptive parents) is used. In this encouraged crossover strategy chooses four individuals or parents in an ergodic manner to produce children. To obtain the best TSP results, we travel from one node to the next while keeping the lowest possible travel cost depending on the TSP condition.

Algorithm 9 Multi-Parent Crossover

Step 1: Randomly generated individuals
Step 2: Calculate a fitness value for each individual
Step 3: while time is not reached do
while time is not reached **do**
repeat
1) perform selection
2) generate population with new individual
end while

5.3.5 Position Crossover(PX)

In the case of Position Crossover[15] What is happening is, firstly random positions in the chromosome are chosen and the selected position point s are settled on the equivalent position in the next chromosome.
Example:

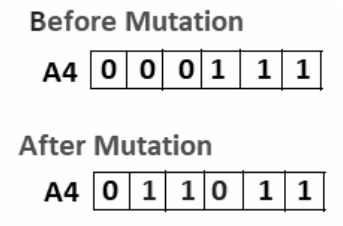
(1 2 3 4 5 6 7 8) and (2 4 5 6 8 7 5 3 1)

By selecting 2nd, 3rd and 6th positions, generated following offsprings:

(1 4 6 2 3 5 7 8) and (4 2 3 8 7 6 5 1)

5.4 Mutation

A genetic operator called mutation is employed to maintain genetic variation from one generation of a population of chromosomes in a genetic algorithm to the next. It's comparable to biological mutation. The following image shows how mutation is done.



Random Mutation

Selection for mutation, Generate a random number r from the range [0, 1] for each p(n) solution. The solution is taken for mutation if r < P_m. In mutation process, To mutate a TSP solution, X = (x_1 , x_2 ,..., x_N) with T number of nodes, choose T number of nodes at random from the solution and simply replace their places in the solution, i.e., if two nodes, x_i , x_j are selected at random, then swap x_i, x_j to get a child solution. The parent solution is replaced by the new solution if it satisfies the problem constraint. For CSTSP to mutate a solution (X,V), where X = (x_1, x_2 ,..., x N), an integer from the range [1, 2] is first selected at random. Two other random integers i, j are selected in the range [1, N] if 1 is selected. Next, swap x_i and x_j to get the child solution. The child solution replaces the parent solution if it satisfies the problem constraint.

5.4.1 Inversion Mutation

The basic inversion mutation[14] [8] picks two cut points in the string at random and reverses the substring between these two cut points. Consider the tour

(1 2 3 4 5 6 7 8 9)

and assume that the first cut point is picked between cities 3 and 4, and the second cut point is chosen between cities 6 and 7.As a result, the offspring produced is

(1 2 3 6 5 4 7 8 9)

Algorithm 10 Inversion Mutation

Step 1: Take an individual as an input.

Step 2: Take two random points (r1, r2) (r1 r2).

Step 3: Make a substring using the genes between the two random points.

Step 4: Invert the substring.

Step 5: Insert the substring in the exact position between the random points.

Step 6: Return the mutated individual.

5.4.2 Insertion Mutation

In insertion mutation (Fogel (1988), Michalewicz (1992))[8], we randomly select cities and remove them from the tour. Those cities are replaced by a randomly chosen location. Consider the tour

(1 2 3 4 5 6 7 8)

and imagine that the insertion mutation operator selects city 3, removes it, and puts it at random after city 6. As a result, the offspring produced is

(1 2 4 5 6 3 7 8)

Algorithm 11 Insertion Mutation

Step 1: Take an individual as an input.

Step 2: Select a random position.

Step 3: Insert the relative gene at a random position and swap the other gene with respect to it.

Step 4: Return the mutated individual.

5.4.3 Exchange Mutation

In exchange mutation [8] [13] pick 2 cities from the chromosome randomly and swap them. Let us consider the tour

(1 2 3 4 5 6 7 8 9)

and assume that city 4 and city 6 are chosen at random.As a result, the offspring produced is

(1 2 3 6 5 4 7 8 9)

Algorithm 12 Exchange Mutation

Step 1: Take an individual as an input.

Step 2: Select a random position (r1, r2) (r1 r2).

Step 3: Swap the relative genes with each other.

Step 4: Return the mutated individual.

5.4.4 Heuristic Mutation

The heuristic mutation selects a multiple number of genes and evaluates all the possible permutations out of them and considers the best. Selected position at random -

(1 2 3 4 5 6 7 8 9)

the off-springs formed with the selected genes -

(1 2 3 4 5 8 7 6 9)
(1 2 8 4 5 3 7 6 9)
(1 2 8 4 5 6 7 3 9)
(1 2 6 4 5 3 7 8 9)
(1 2 6 4 5 3 7 8 9)

Algorithm 13 Heuristic Mutation

Step 1: Take an individual as an input.

Step 2: Select n number of random positions.

Step 3: Pick up relatively n number of genes.

Step 4: Generate individuals according to all possible permutations of the selected genes.

Step 5: Evaluate all individuals and select the best one.

Step 6: Return the best selected individual.

6 Result

We tasted our model on multiple benchmark instances and got satisfactory results.

The below represents the cost table of 4D TSP.

This is a 4D data matrix of 5 cities, 3 roads and 2 vehicles.

		0		1		2		3		4	
		v1	v2	v1	v2	v1	v2	V1	V2	V1	V2
0	r1	0	0	73	7	58	88	95	91	18	82
	r2	0	0	50	4	29	5	8	10	51	38
	r3	0	0	10	61	8	47	46	72	81	22
1	r1	73	7	0	0	96	47	45	59	3	79
	r2	50	4	0	0	9	18	80	55	81	85
	r3	10	61	0	0	57	6	74	27	17	96
2	r1	58	88	96	47	0	0	93	64	82	24
	r2	29	5	9	18	0	0	68	27	53	79
	r3	8	47	57	6	0	0	10	11	31	94
3	r1	95	91	45	59	93	64	0	0	62	67
	r2	8	10	80	55	68	27	0	0	55	86
	r3	46	72	74	27	10	11	0	0	49	52
4	r1	18	82	3	79	82	24	62	67	0	0
	r2	51	38	81	85	53	79	55	86	0	0
	r3	81	22	17	96	31	94	49	52	0	0

Population Size = 200
Elitism Rate = 0.2
Mutation Rate = 0.05
Crossover Rate = 0.6
Generation No = 500

We tried to solve 4D TSP, and after several tries, the minimum cost we got for this Tsp is 54.

Choose some benchmark problems and shown our obtained results in below table.

Population Size = 300
Elitism Rate = 0.2
Mutation Rate = 0.05
Crossover Rate = 0.6
Generation No = 300

Roulette Wheel Selection, Ordered Crossover and Exchange mutation are used in these trials.

Result		
Benchmark TSP Problems	Optimal Cost	Obtained Cost
five	19	19
p01	291	291
gr17	2085	2085
dantzig42	699	713
fri26	937	962
att48	33523	34494

Population Size = 200
Elitism Rate = 0.2 Mutation Rate = 0.05 Crossover Rate = 0.6 Generation No = 500

Comparison of Performance of different Algorithms Used in GA for TSP :

In the upper sections we already discussed many algorithms that are used in various aspects of Genetic Algorithm like selection, mutation etc.
We conducted a performance survey between a few of popular algorithms that are used widely in this field.
In order to do that we took the "ch130" benchmark TSP, we will try to solve this problem with all algorithms separately, and then we will compare their results.

6.1 Selection Algorithms comparison:

popSize = 300
elitismRate = 0.2
mutationRate = 0.05
crossOverRate = 0.6
generationNo = 100

Crossover algorithm used Ordered Crossover and Mutation algorithm used Exchange Mutation.

Selection Algorithms comparison table		
Algorithm Name	Obtained Cost	Time taken
Roulette Wheel Selection	24917.0	3.24
Tournament Selection	21252.0	4.03
Random Selection	24191.0	2.9
Rank Selection	21440.0	3.2

6.2 Crossover Algorithms comparison:

popSize = 300
elitismRate = 0.2
mutationRate = 0.05
crossOverRate = 0.6
generationNo = 100

Selection Algorithm used Roulette Wheel Selection and Mutation algorithm used Exchange Mutation.

Crossover Algorithms comparison table		
Algorithm Name	Obtained Cost	Time taken
Ordered Crossover	24941.0	3.21
Cyclic Crossover	31282.0	2.72
Partially Mapped Crossover	26778.0	6.67
Position Based Crossover	29484.0	5.94

6.3 Mutation Algorithms comparison:

popSize = 300
elitismRate = 0.2
mutationRate = 0.05
crossOverRate = 0.6
generationNo = 100

Selection Algorithm used Roulette Wheel Selection and Crossover algorithm used Ordered Crossover.

Mutation Algorithms comparison table		
Algorithm Name	Obtained Cost	Time taken
Inversion Mutation	24146.0	3.27
Insertion Mutation	23423.0	3.39
Exchange Mutation	25388.0	3.27
Heuristic Mutation	24913.0	3.57

Conclusion

In this literature we provide the overview of Genetic Algorithm as well as Travelling Salesman Problem. The TSP seems Like a very simple task but it is very tough to resolve. The fact is find the shortest travel path among the all possible path. NP-hard problem like TSP cannot be solved in polynomial time when the points are so large in number. Many algorithms have been developed in this case. We looked at a variety of representations and operators that could be employed in evolutionary algorithms to solve the Traveling Salesman Problem. The path representation is used in this study to describe the representation. It is also the most often used, and a huge number of operators have been created for it.

Although it was outside the scope of this paper, we briefly explored genetic algorithms and their various techniques. We did this since the development of a good evolutionary algorithm appears to be unavoidable. A comparison of the results achieved with the approximations based on the Genetic Algorithms examined here with the results obtained with the other techniques mentioned in the preceding section could be of interest.

References

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

[2] Determination of Selection Method in Genetic Algorithm for land stability, Department of Information System, Faculty of Industrial Technology University of Pembangunan Nasional "Veteran"Jawa Timur, Surabaya, Indonesia, Department ofComputer Science and Electronic Faculty of Mathematic and Natural Science, Gadjah Mada University, Yo gyakarta,Indonesia Faculty of Agriculture Gadjah Mada University, Yogyakarta Indonesia.

[3] Genetic Algorithm Performance with Different Selection Strategies in Solving TSP. Noraini Mohd Razali, John Geraghty. Review Paper.

[4] Genetic Algorithms for the Travelling Salesman : A Review of Representations and Operators. January 1999, DOI:10.1023/A:1006529012972,Source DBLP. Pedro Larranaga-Universidad Polit´ecnica de Madrid, Cindy Kuijpers-Tilburg University.

[5] International Journal of Advanced Research in Computer Science and Software Engineering. Solving Travelling Salesman Problem Using Genetic Algorithm. Saloni Gupta, Poonam Panwar Department of Computer Science Engineering Ambala College of Engineering Applied Research, Ambala- 133101, India.

[6] Introduction to Algorithm, by S.N.Sivanandam · S.N.Deepa.(Ref. Book)

[7] Literature Review on Travelling Salesman Problem, Chetna Dahiya Shabnam Sangwan Department of Computer Science Sat Kabir Institute of Technology and Management, Maharishi Dayanand University(MDU),Rohtak.https://www.researchgate.net/publication/341371861

[8] Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators P. LARRANAGA, C.M.H. KUIJPERS, R.H. MURGA, I. INZA and S. DIZDAREVIC Dept. of Computer Science and Artificial Intelligence, University of the Basque Country, P.O. Box 649, E-20080 San Sebastian, The Basque Country, Spain.

[9] Traveling Salesman Problm Using Genetic Algorithm. Mohammad Asim-Assistant Professor, Department of Computer Science MGM College of Engineering and Technology, Noida, Uttar Pradesh, India.http://se-ssuet.move.pk/wp-content/uploads/2017/07/Travelling-problem.pdf?i=1

[10] S. Samanta, A. De and S. Singha, "OLUTION OF TRAVELING SALESMAN PROBLEM ON SCX BASED SELECTION WITH PERFORMANCE ANALYSIS USING GENETIC ALGORITHM," International Journal of Engineering Science and Technology, vol. 3, no. 8, 2011.

[11] J.-Y. Potvin, "Genetic algorithms for the traveling salesman problem," Annals of Operations Research, vol. 63, no. 3, pp. 337–370, 1996.

[12] I. Korejo , S. Yang and C. Li, "A directed mutation operator for real coded genetic algorithms," in European Conference on the Applications of Evolutionary Computation, 2010.

[13] W. Banzhaf, "The "molecular" traveling salesman," Biological Cybernetics, vol. 64, no. 1, pp. 7-14, 1990.

[14] D. A. Fogel, "A parallel processing approach to a multiple travelling salesman problem using evolutionary programming," in Proceedings of the Fourth annual Symposium on Parallel Processing, 1990.

[15] CROSSOVER OPERATORS IN GENETIC ALGORITHMS: A REVIEW A.J. Umbarkar and P.D. Sheth Department of Information Technology, Walchand College of Engineering, India

[16] H. Muhlenbein, M. Gorges-Schleuter, and O. Kramer, "Evolution Algorithms in Combinatorial Optimiz

[17] CROSSOVER OPERATORS IN GENETIC ALGORITHMS: A REVIEW A.J. Umbarkar and P.D. Sheth Department of Information Technology, Walchand College of Engineering, India.

[18] K. F. Man, K. S. and Tang, S. Kwong, (201), Springer, "Genetic Algorithm Concepts and Designs."

[19] Mrs. Geetha Ramani.R, Nishaa Bouvanasilan,Vasumathy Seenuvasan, " A perspective view on Travelling Salesman Problem using Genetic Algorithm".

[20] A Study of Genetic Algorithm and Crossover Techniques Ashima Malik Assistant Professor,Computer Science Dept., Dronacharya College of Engineering, Gurugram, India.

[21] P. J. Bentley and J. P. Wakefield, "Hierarchical Crossover in Genetic Algorithms",Proceedings of the 1st On-line Workshop on Soft Computing, 1996.

[22] Kengo Katayama and Hiroyuki Narihisa, "An Efficient Hybrid Genetic Algorithm for the Traveling Salesman Problem", Electronics and Communications in Japan.

[23] Sengoku, H., Yoshihara, I., "A Fast TSP Solution using Genetic Algorithm (Japanese)",Proceedings of 46th National Convention of Information Processing Society of Japan, 1993.

[24] S. Tsutsui and A. Ghosh, "A study on the effect of multi-parent recombination in real coded genetic algorithms", Proceedings of IEEE World Congress on Computational Intelligence.

[25] Chilukuri K. Mohan, "Selective crossover: towards fitter offspring", Proceedings of the ACM symposium on Applied Computing.

[26] B. Van Gucht, D. (1985). Genetic Algorithms for the TSP. In Grefenstette, J. J.(ed.) Proceedings of the First International Conference on Genetic Algorithms and Their Applications, 160–165. Hillsdale, New Jersey: Lawrence Erlbaum.

[27] Homaifar, A. Guan, S. (1991). A New Approach on the Traveling Salesman Problem by Genetic Algorithm. Technical Report, North Carolina AT State University.