NUS National University of Singapore | ISS INSTITUTE OF SYSTEMS SCIENCE

KE5207 – COMPUTATIONAL INTELLIGENCE II

Public Transport Route Optimization using Evolutionary Algorithm

Submitted by:

Anjali Sinha - A0178476L

Gopalakrishnan Saisubramaniam – A0178249N

Madan Kumar Manjunath – A0178237W

Viknesh Kumar Balakrishnan – A0178304E

Yesupatham Kenneth Rithvik – A0178448M

Table of Contents

Introduction

In order for any city or country to develop and progress in the modern world, the public transport system usually proves to be one of the most vital aspects of this endeavour. Connecting cities together within a country, effectively brings together the people residing in these cities so that they can collaborate, work and socialise together. In this project, we aim to use optimization techniques, in particular evolutionary algorithms to plan a near optimal set of routes that our public transportation system can use in order to increase ridership as well as reduce the cost of operation. In order to solve this problem we have come up with a dataset of 25 cities and the distances between these cities as well as the population or average ridership of these cities.

Problem Description

The task of finding the most optimal route between a set of cities based on the distance between them can be modelled as a variation of the travelling salesman problem.

Travelling Salesman Problem

This problem basically tries to find the shortest route that traverses all the possible locations and reaches back to the original source location. This is a NP-hard problem. The time required to find the best solution to this problem is non-deterministic and is usually a lot as the number of cities increases.

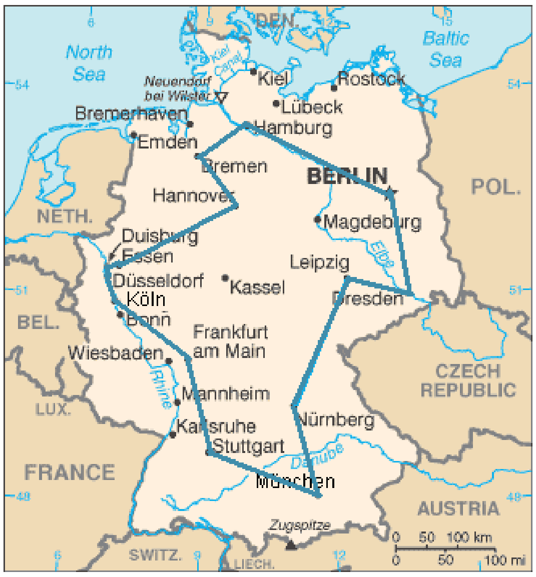

Figure 1 Most optimal TSP route for 15 cities in Germany

This problem cannot be solved in a practical period of time for a large number of cities. Therefore most of the solutions to this problem seek to employ combinatorial optimization to get the closest to best solution.

Application of TSP in our Problem

Some of the modifications that has to be incorporated into the travelling salesman problem in order to solve our problem can be summarized as follows:

- The route traversed does not have to end at the same location as the start location. In fact the start and end point can be wide apart from each other for a public transport route.

- One of our goals is to find the shortest possible route but this is not our most important constraint.
- Our most important optimization task is to find the route that can pass through the cities with the most number of people.
- We also employ a fuel constraint to the routes planned where we set a max distance for which a vehicle can travel without re-fuelling.
- Our problem requires us to find multiple diverse routes for our defined set of cities instead of just one route.

From the above points it can be seen that our problem needs us to optimise multiple parameters in order to reach our solution instead of just finding the shortest route possible.

Proposed Solution

From the description of the problem we can see that we have to employ an optimization procedure to come up with a near best solution. It is not possible to determine the best set of routes in a deterministic amount of time.

Therefore we employ the evolutionary algorithm approach to model our problem. We believe that it is a right fit to solve this problem as the genetic algorithm can come up with multiple routes and sub-routes that can later be re-combined in multiple combinations to arrive at an optimal solution.

The basic set of steps that we employ to model this problem are:

- We collect the data of all cities, the approximate distance between them, the population or ridership in each city and the fuel depot availability in each city.
- Create an initial set of routes that will serve as the starting point of our application. These routes can be formed using basic heuristics, such as the links between cities that are the shortest, including the cities that have the largest population, etc.
- Come up with a method to evaluate the goodness of each set of routes obtained. This metric can be maximised or minimized to obtain the optimal result. In our problem, this method needs to maximise ridership while minimising the total distance of the route. We also need to check the feasibility of the route by setting a threshold for the maximum number of cities a vehicle can travel before needing to reach a fuel depot to re-fuel.
- Come up with different strategies that employ genetic operators to modify the set of routes obtained during each generation.
- Determine when to stop the evolutionary process either based on the goodness measure or based on a predefined set of generations.

The proposed solution is represented with a flowchart in the below diagram.

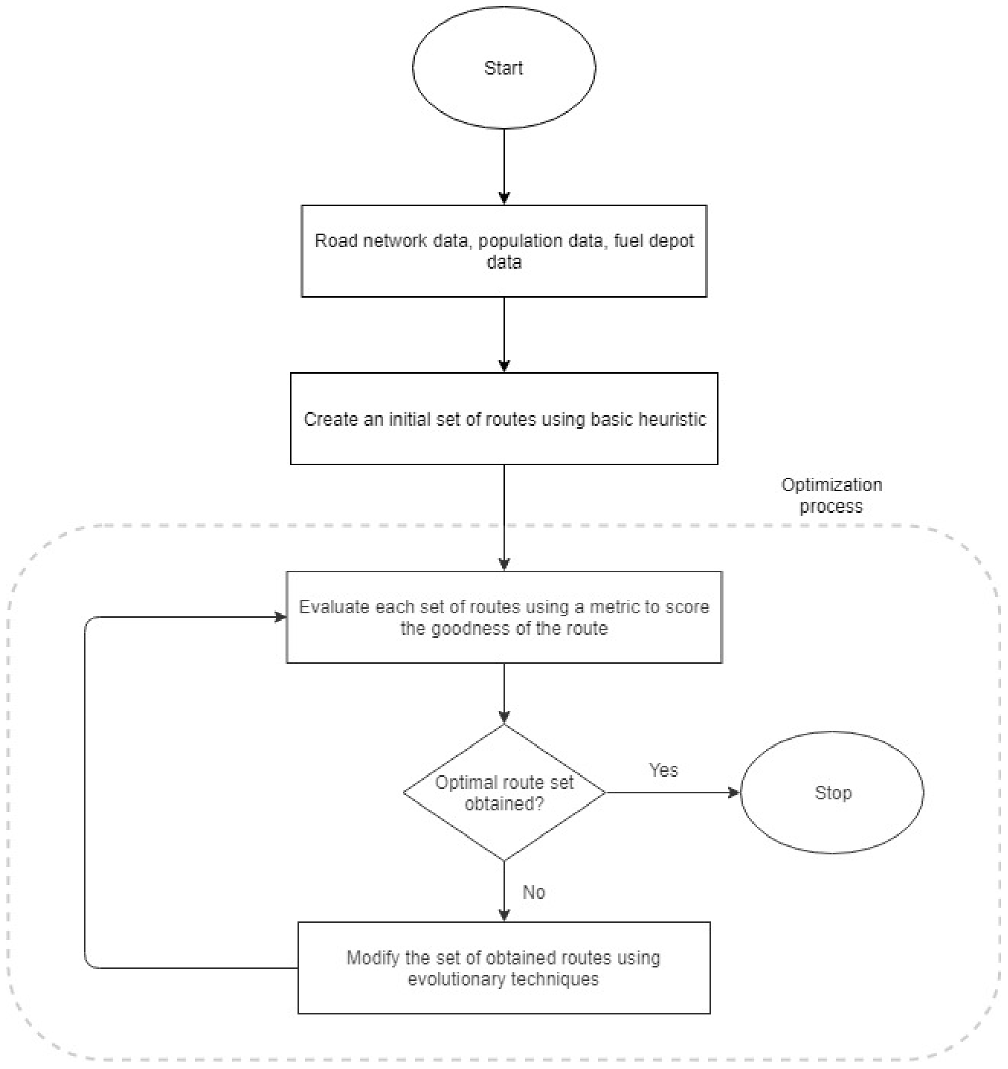
Start
Road network data, population data, fuel depot data
Create an initial set of routes using basic heuristic
Optimization process
Evaluate each set of routes using a metric to score the goodness of the route
Optimal route set obtained?
Yes
Stop
No
Modify the set of obtained routes using evolutionary techniques

Figure 2 Flowchart of proposed solution

Genetic Algorithm

Genetic Algorithm is a popular evolutionary algorithm approach to solve optimization problems in various applications. This algorithm was developed from biological concept which involves utilizing parents and produce offspring through crossover, mutation and selection of best individuals. Here in this problem, we wanted to select the best routes between cities which can optimize the fuel utilization, distance and time travelled. In this process, we must decide on the structure of the chromosome, generate random initial population and define appropriate fitness function to obtain better chromosome in every generation and hence the best optimized routes.

Data

To obtain best travel routes from the route optimizing system, it must be provided with appropriate data to assist in the selection of the best routes. We had two data sources as mentioned below:

I. City Details:

1. City – 25 cities

2. Population of each city
3. Availability of Petrol Pumps – 11 cities have petrol pumps

II. City Matrix: Distance matrix between each of the 25 cities

Majority part of the data comes in the fitness function definition and optimizing the results to obtain the best results.

Chromosome

To build the structure of the chromosome, initial requirement or heuristics is needed from the business. In this case, some requirements like number of routes for the region/city/country and number of cities to be covered in a route. In this route optimizing problem, the number of routes and number of cities in each route is assumed to be 10 and 8 respectively. Hence a chromosome will contain 10 different routes and each route connecting to 8 different cities.

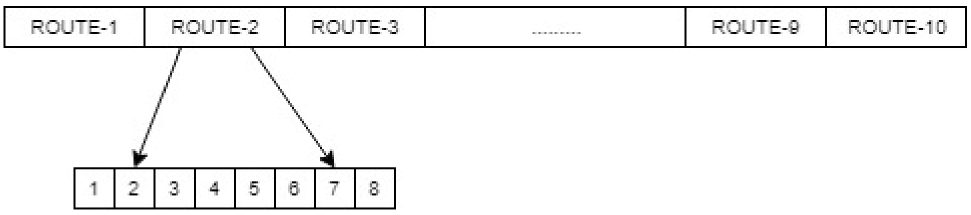

Figure 3: Chromosome Structure

Initial Population

The idea behind creating the initial population is that to give a start point for the whole search process by finding the next best solution from the initial set. The size of the population is decided as 100. To start with, the parents in the initial population are selected randomly but made sure satisfying some basic conditions like each route consists of 8 non-overlapping cities, each parent consist of 10 routes and covering all the given cities. Since this is built random, it provides the freedom for creating few parents with heuristics from the business or SME's. Initial population can also be set to start from a different highly populated city in each route of parent.

Fitness Function

For any GA optimizing problem, it is required to construct a fitness function to evaluate each parent chromosome and select best parents to create the offspring for the next generation and so on until the best result. Evaluation of a single route doesn't cover the whole picture of the optimization here and hence all the routes in a route set or parent should be considered while calculating the fitness value. The fitness values are calculated based on the below criteria to measure the goodness of each parent or route set.

1. All cities in the given region should be covered at least once in any given route set or parent
2. Minimising the distance travelled in every route and hence the time as well
3. Fuel constraint of the vehicle
4. Total Population covered by every route set or parent

Several other evaluation metrics can be appended to the former list depending upon on the business requirement and obtain relevant best results. Above evaluation metrics constructing the fitness function are explained further below.

Fitness Function = City Coverage + Distance Travelled + Fuel Constraint + Population Covered

City Coverage

Main aim of any public transport system would be to establish routes connecting every potential city. Hence, this was framed as the first fitness function to evaluate if all the given cities are covered in a single route set. With above said, this function ensures every city occurs at least once in the route set and more importantly non-overlapping of cities within a route. Anyways, Non-overlapping of cities is defined in the fitness function, but it is taken care in procedures like creating the initial set of population or crossover or mutation. This evaluation metric does not have any numeric value but a binary output of True or False.

Distance Travelled

City Matrix data file provides the distance between each city which can be used to calculate the distance travelled in any route or route set. Basically, this is a calculated value that will be one of the contributions to the final fitness value. Here, a constraint was set that the any single route cannot travel beyond a limit. This is to ensure that we do not get a single route which comprises of all largely distant city as it will be wastage of resources like human, fuel or time. The idea behind this fitness function is that to minimize the total distance travelled by all the routes in a route set and get maximum coverage, the bigger picture of the latter portion is yet to be explained and once the other fitness functions are well explained, it will provide a wholistic idea about the design of the fitness function.

Fuel constraint

Fuel constraint is an important factor in any public transportation system. It was prefixed that every vehicle can only travel distance limit with full level of fuel. The importance or need for this fitness function is that it provides reliability of the output from the system. For example, without fuel constraint, possibility of getting a best route with some distant cities in route is high which in real case is not possible without refuelling the vehicle. From the city details file, we would get the information whether the city has a petrol pump station or not. The output of this fitness function would also be binary output of feasible route or not. Here again, we calculate the distance travelled from first city to the last city in a single route one by one and reset the distance travelled back to zero when a city has petrol pump. If the vehicle happens to travel beyond the maximum distance of the petrol capacity without reaching to a city with petrol pump, the route is considered as not feasible. This fitness function gives a great boost to the reliability of the output from the system as said earlier.

Population Covered

Finally, the very important factor in the total fitness value is the total population covered by all individual routes in the route set. The idea is to maximize this fitness value so that more people in that get benefitted by the transport services. This is very common factor for any public services offered by the country as the motive would be reach to the at most population. City details file provides us with details about the population of each city which is used in this fitness function to obtain the overall population covered by the route set. This fitness function ensures that cities with less distance are not connected to form a route in the route set which was driven the previous fitness function but to maximise the population covered as well. Below picture shows the existing public transportation route in Colorado where multiple routes tend to cover the most populated places in the region.

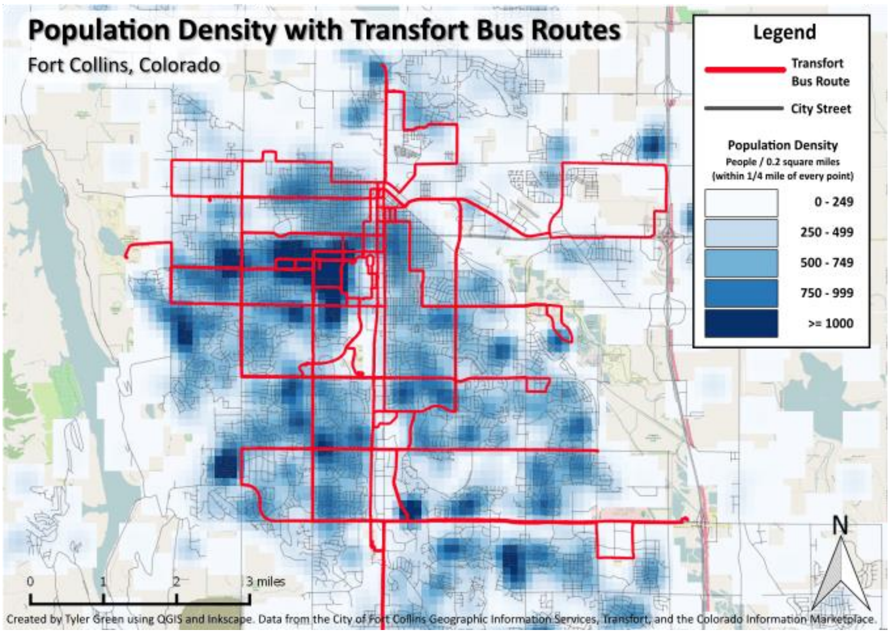
Population Density with Transfort Bus Routes
Fort Collins, Colorado
Legend
Transfort Bus Route
City Street
Population Density
People / 0.2 square miles (within 1/4 mile of every point)
0 - 249
250 - 499
500 - 749
750 - 999
>= 1000
N
0
1
2
3 miles
Created by Tyler Green using QGIS and Inkscape. Data from the City of Fort Collins Geographic Information Services, Transfort, and the Colorado Information Marketplace.

Figure 4: Densely populated area covered by several routes

Hard and soft Constraints

Every optimization problem in the real world has to find a solution under some constraints or restrictions. These constraints are usually of two types- hard and soft.

Hard Constraints

The hard constraints have to be met at any cost and form an important part of a feasible solution. These constraints cannot be violated or else the solution itself is non-feasible or does not exist in the real world. In our solution, the hard constraints that we have employed as well as mentioned before are:

- Only consider routes where the maximum distance between cities with a fuel depot is 200 km
- Never travel to cities that have already been visited before in a route, or eliminate cycles in a route
- The length of the route should always be 8 cities

Different methods can be used to enforce hard constraints. In our case we remove infeasible solutions.

Soft Constraints

These types of constraints are regarded as a lot more flexible while optimising our genetic algorithm to find a solution. A small violation in the soft constraint is acceptable as long as it gives a more optimal solution. Solutions that violate soft constraints are still considered to be feasible solutions. The soft constraints in our problem are:

- The max distance of any particular route should be less than 300 km
- All the 10 routes that we find should be as different as possible.
- The routes should cover all 25 cities.

To employ these soft constraints, we penalize our fitness function to give lesser values when these constraints are violated.

Crossover

The generic definition of crossover is to exchange portion of the better features of each parent to produce better offspring. Here each route set with 10 different routes is considered as parent and we need to cater the crossover between them to produce different efficient route sets which can be implemented in two different ways depending upon the scope. The crossover is controlled by the usage of crossover rate which decides whether crossover should happen between given two parents.

Inter crossover

Inter crossover is defined as the exchange of some complete routes between two different route sets. This can also be called as Route set modification as shown in the figure below as exchange happens between route sets. For better understanding let us consider two parents with 5 routes each. As per the crossover operator, the cut points are chosen randomly, and the parents are split into parts based on the cut points. Later parts from each parent are exchanged and form a new offspring. In the below example cut point is chosen as 2 and hence routes 3,4,5 from each parent are exchanged as shown in the table. We have not kept any constraint on the produced offspring and the fitness of the offspring would be evaluated at the later stage and if it fails to meet soft constraints it would be assigned a very low fitness score so that it would never feature in the population of subsequent generations.

Parent #1	*BNXTQEAK,THNSXLDQ,QDXTUECN,QEXTFUIN,PNXTLDGQ*
Parent #2	QXDWNTOE,NSYETQXF,XQNETDUB,TXQJHNEL,EUNCVQXT
Offspring #1	*BNXTQEAK,THNSXLDQ,*XQNETDUB,TXQJHNEL,EUNCVQXT
Offspring #2	QXDWNTOE,NSYETQXF,*QDXTUECN,QEXTFUIN,PNXTLDGQ*

Intra crossover

Intra crossover is defined as the exchange of portion of a route between two different routes in a single route set. This can also be called as Route modification as shown in the figure below as exchange happens between routes in a route set. The differential portion in this crossover is that multiple copies of the population is generated, say 'm' because the routes for crossover is also selected randomly. So, with multiple copies of same population, this crossover could produce different crossover offspring from a single parent.   To start with the procedure, two routes in the parent and the cut points are chosen randomly to perform crossover. In the below example cut point is chosen as 4 for crossover, route 4 & 5 are chosen for crossover to produce offspring #1, route 1 & 5 are crossover to produce offspring #2 and route 2 & 3 are crossover to produce offspring #3 which is shown in the below table as well.

Parent	BNXTQEAK,THNSXLDQ,QDXTUECN,QEXTFUIN,PNXTLDGQ
Offspring #1	BNXTQEAK,THNSXLDQ,QDXTUECN,QEXT**LDGN**,PNXT**FUIQ**
Offspring #2	**PNXT**QEAK,THNSXLDQ,QDXTUECN,QEXTFUIN,**BNXT**LDGQ
Offspring #3	BNXTQEAK,THNS**UECN**,QDXT**XLDQ**,QEXTFUIN,PNXTLDGQ

Mutation

Mutation in genetic algorithm provides light variation to the offspring by flipping any gene. This is important as it can provide a greatly improved offspring. Mutation is not advisable on all the offspring and it should be controlled by mutation rate which decides upon the mutation. At initial stage, it is good to have a lesser mutation and at later stage when the result is not varying much, we can increase the mutation rate to accelerate the movement towards global minima or maxima. Here, mutation rate is considered as 7% and each route in the route set goes through this mutation procedure but based on mutation rate, flipping of a city in the route happens. In the below example, we can see that only mutation has happened only in two routes and other 3 routes are unchanged after the mutation process.

Parent	BNXTQEAK,THNSXLDQ,QDXTUECN,QEXTFUIN,PNXTLDGQ
Offspring	BNXT***V***EAK,THNSXLDQ,QDXTUECN,QEX***S***FUIN,PNXTLDGQ

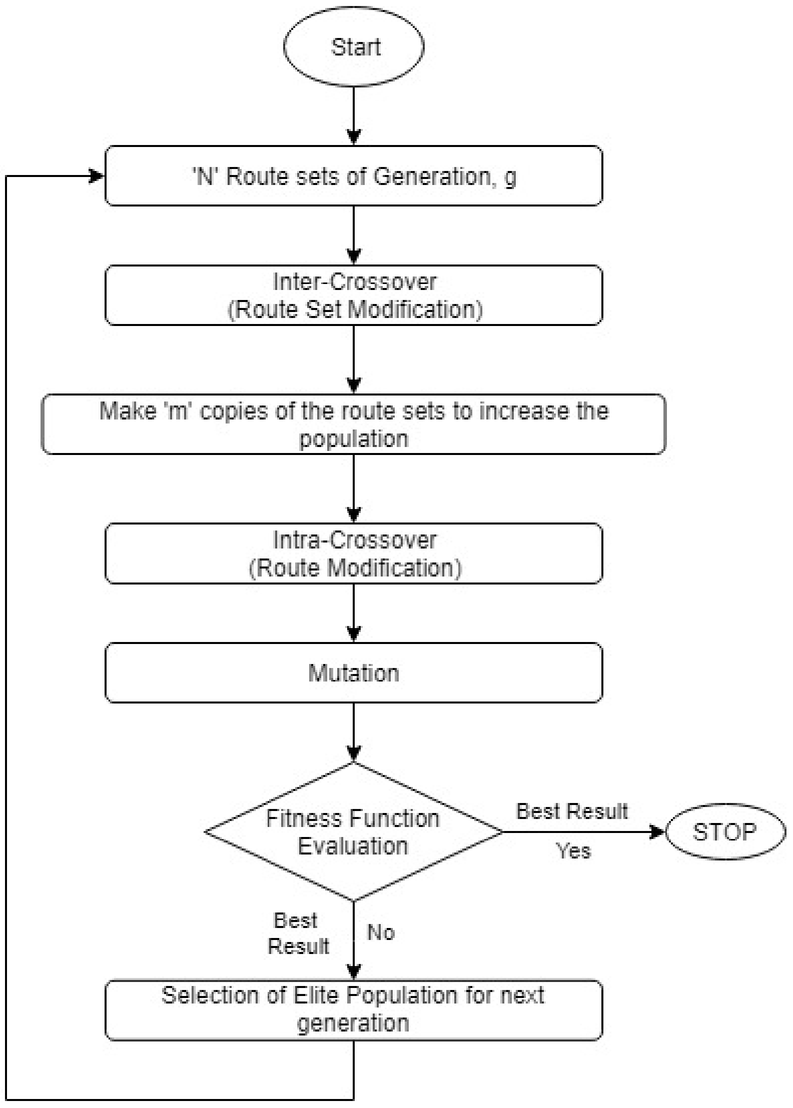
Start
'N' Route sets of Generation, g
Inter-Crossover (Route Set Modification)
Make 'm' copies of the route sets to increase the population
Intra-Crossover (Route Modification)
Mutation
Fitness Function Evaluation
Best Result
Yes
STOP
Best Result
No
Selection of Elite Population for next generation

Figure 5: Flowchart of the Genetic Algorithm process for Route optimization

Elite Population

As you can see here, in this experiment of obtaining an optimizing transport route set, it was started with an initial population of 100 chromosomes.

- But in the due process, the initial set has gone through the inter-crossover stage to get 100 new offspring.
- Rejection of any chromosome should purely happen through the fitness function and hence, the initial population and the inter-crossover offspring are combined and sent to the next stage of intra-crossover.

- As mentioned earlier, to get more varieties of offspring from the intra-crossover stage, the grouped population from the previous step is made into 3 copies. From the intra-crossover stage, we would get 300 new offspring which is combined with inter-crossover and initial population and presented to the mutation stage.
- Depending upon the mutation rate, mutation happens on each chromosome and finally we get a pool of 600 chromosomes which is sent to the evaluation stage where the fitness values for each of chromosome is calculated.
- Now based on the fitness value, top 100 chromosomes are selected for next generation and goes through same procedure until the best result is obtained.

GA Parameter Optimization

While running the genetic algorithm multiple times to find an optimal solution, we tried changing our hyper-parameters to get better values from our fitness function. The final hyper-parameters that we choose, and their value are given below:

Parameter	Range	Value
Number of Cut-points	1,2	2
Mutation rate	0.01 to 0.1	0.03
Population Size	100 to 600	600
Elite population	-	100
Generation gap	-	0.833
Generations	100 to 500	100

Results

As a result of 100 generations of evolution from the initial population, we have obtained the below as the best route set which has below salient features:

- Covers all the 25 cities in 10 routes
- No overlapping of cities within a route
- Fuel constraint is also taken care
- Most populated cities like 'Q', 'N', 'X', 'T', 'E' can be seen appearing multiple times in the route set
- Total distance travelled in all 10 routes is minimised (from the table & figure-7)
- Total Population covered in all 10 routes is maximised (from the table & figure-6

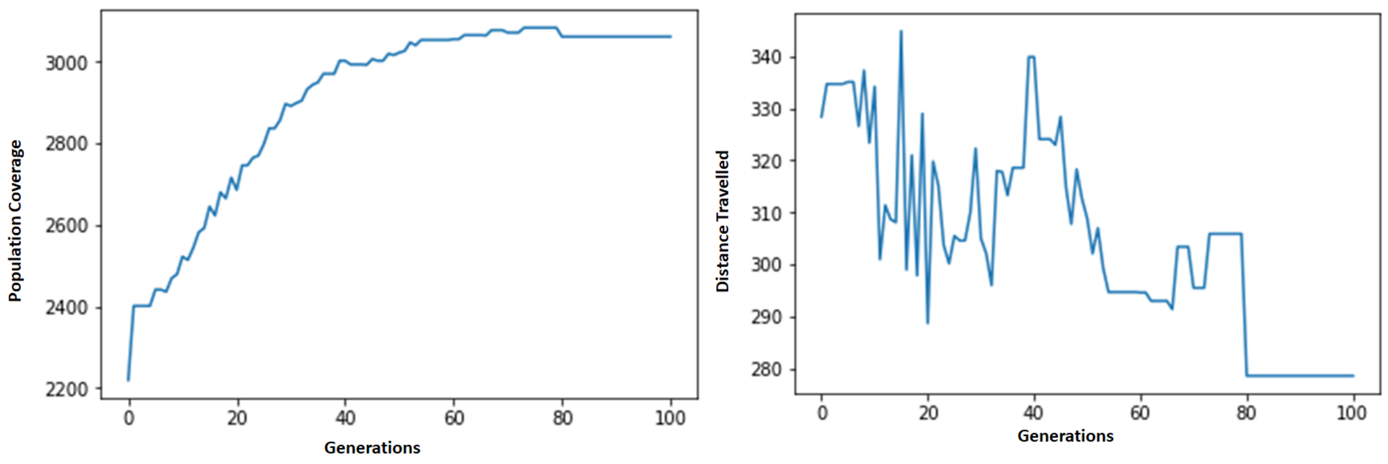

Figure 6: Population coverage and distance travelled over generations

Route #1	N→A→E→Q→R→T→X→J
Route #2	E→A→K→T→X→D→Q→N
Route #3	N→A→E→Q→R→T→X→D
Route #4	U→D→Q→E→N→X→T→R
Route #5	Q→S→C→A→N→X→E→V
Route #6	N→A→E→Q→X→Y→M→D
Route #7	U→D→Q→E→N→X→H→T
Route #8	N→I→X→T→U→D→Q→E
Route #9	L→D→U→G→Q→N→X→B
Route #10	T→O→F→Q→W→P→N→X
Population Coverage	3062
Distance Travelled	2786
Fitness Value	2783.4

Table 1: Best Route set from the GA optimization Process

- From the Population coverage graph, we can see that population coverage fitness function has played a major role in getting the best route set as it has seemed to improve over the generations and flattened out towards the end.
- From distance travelled graph, we can see that the distance travelled could not get the smooth decline because of the crossover and mutation happened in the due process but eventually towards the end, it was able to find the best combination of routes that would reduce the total distance travelled in a route set.
- As the fitness function was defined as maximising population coverage and minimising the distance travelled, the fitness value graph also looks similar to the population coverage graph

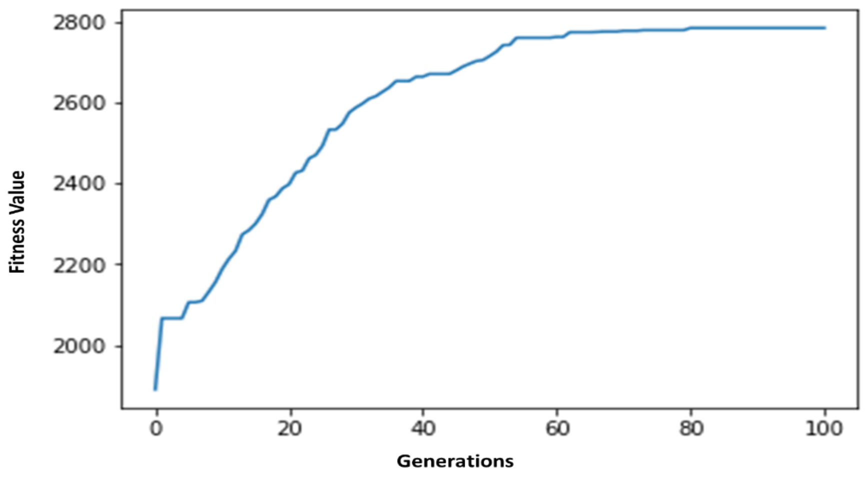

Figure 7: Fitness value comparison over generations

Conclusion

While carrying out this assignment, we were able to get a practical understanding approach of how evolutionary algorithms work and how they can be used to model real world problems. From the techniques used to model this problem and subsequently, the results that were obtained we can say that we have got a set of 10 optimal routes to form a public transport system that covers quite a significant part of the population. It would have been a tedious process to do this manually while taking all the constraints into consideration. It would have taken a non-deterministic amount of time to solve this problem using conventional computer algorithms that are programmed to find the best route possible. Considering all this, we believe genetic algorithms are a very good approach to automate this process and get results in an acceptable amount of time.