CS 451: Computational Intelligence

Assignment 01

Syed Hasan Faaz Abidi (sa06195), Syed Hammad Ali (id daalo), Hasan Naseem (id daalo)

13 February 2022

1. TSP Problem

As required by the assignment, we had to optimize the Traveling Sales Person problem on the Qatar dataset containing 194 cities using Evolutionary Algorithm.

For this particular problem, the chromosome representation was as follows,

1.1. Chromosome Representation

In the TSP dataset, there were 194 nodes representing each city with their position in euclidean distances. Keeping the constraints in consideration of a valid route that the route visits each city exactly once and returns to the origin city, the representation we chose was a simple **1D list**. Each element of the list represents a unique integer for each city.

For example if we have 3 cities, a possible candidate solution with population 3 looks like:

Population = [[1,2,3],[3,2,1],[2,3,1]]

1.2. Fitness Function

The fitness function for this problem was straightforward. The fitness score in TSP for a valid route/solution was the total distance between each city in that route. So, to calculate the total score, we first calculated the distance between each city using the given euclidean distances in the data. Summing up all those distances would give us a fitness score of a particular route.

TSP was different in a way that to optimize it, we need to minimize the fitness score. However, our selection functions were implemented in a way that they were biased towards the fittest chromosome. To counter this issue, we made our fitness values negative, so that selection would work the same but TSP would also get optimized.

Pseudocode for fitness function:

```
def __fitness_function(self, chromosome: list)-> int:
        total_distance = 0
        for i in range(len(chromosome)-1):
                total_distance += self.__distance(chromosome[i],
                chromosome[i+1])
        total_distance += self.__distance(chromosome[0], chromosome[-1])
        return total_distance*-1

def __distance(self, city1: int, city2: int)-> int:
        return ((self.data[city1][0] - self.data[city2][0])**2 + (self.data[city1][1] -
        self.data[city2][1])**2)**0.5
```

1.3. EA Evaluation

1.3.1 Binary Tournament & Truncation Plots