CS214-Lab 2 Report

Kushagra Khatwani (190020021), Suryavijoy Kar (190020039)

Methodology:

To solve the TSP problem, we have implemented a “2 opt” algorithm. The algorithm is as follows:

- Start with a random tour
- Select any 2 edges of the tour. Reconnect them with each other to form another tour and calculate the new cost of the tour. If this modification has led to a smaller cost, then the current tour is updated.
- The algorithm continues to build on the improved tour and repeats the steps.
- This process is repeated until no more improvements are found or a certain threshold of improvement is reached.

Pseudo Code:

```
def two_opt(self, thr = 0.01):
    self.best_route = self.init_route
    self.best_dist = self.calc_path_dist(self.distances, self.best_route)

    alpha = 1

    while alpha > thr:
        prev_dist = self.best_dist
        for city1 in range(self.n_cities - 1):
            for city2 in range(city1+2, self.n_cities-1):
                edge1_start = self.best_route[city1]
                edge1_end = self.best_route[city1+1]
                edge2_start = self.best_route[city2]
                edge2_end = self.best_route[city2+1]

                dist1 = self.distances[edge1_start][edge1_end] +
self.distances[edge2_start][edge2_end]
                dist2 = self.distances[edge1_start][edge2_start] +
self.distances[edge1_end][edge2_end]
```

```
                if dist2 < dist1:
                    new_route = self.swap(self.best_route, city1, city2)
                    new_distance = self.calc_path_dist(self.distances, new_route)
                    self.update(new_route, new_distance)

        alpha = 1 - self.best_dist/prev_dist
    return self.best_route, self.best_dist
```

Inference:

The stopping criteria we have implemented is thresholding in nature. After every swap, we calculate the ratio of (improved cost of tour)/(previous cost of tour). Initially, this value will be much less than 1 but as we keep on improving the tour, it approaches closer and closer to 1. When it crosses the threshold, we assume that not much improvement is taking place and we can exit our algorithm.

We have tried various values of thresholds and have selected the ones which we found to best fit the euc_100 and noneuc_100 data.

The time complexity is quadratic in the number of cities.

Conclusion:

Having a static threshold for the stopping criteria has its drawbacks. In the initial swappings, the improvements in tour cost is much more than the later swappings. So the threshold value must be adjusted to account for this behavior to prevent early exiting from the loop.