AI Lab-3 Report

Group Number 2 (210010046, 210020047)

Introduction:

In this lab, we were supposed to find the shortest path of the TSP problem on the dataset we presented. We used Ant-Colony-Optimization(ACO).

Motivation:

At first, we were using the Genetic Algorithm to find the shortest path, but the problems we faced were the following:

- The algorithm took a lot of time on a larger population to find the optimal solution for the problem.
- The algorithm was also relatively slow regarding its efficiency and computational speed. We had to reduce the number of new children that we returned or had to begin with a smaller population.
- There were problems finding optimal Mutation functions to make the mutations helpful rather than random swaps.

After that, we switched to ACO and PSO(for larger datasets). These were better than GA in the following parameters:

- There was no need to find the optimal Mutation function, and there was more exploration than exploitation.
- The algorithms were faster than those used in G.A.(s).

Iterative Improvements:

Genetic Algorithms:

We used the GAs to find the optimal path for the TSP. We were using a set of 20 routes for the crossover and were using the Partial Crossover technique for that.

We chose Partial Crossover because it provided an easy way to deal with duplicates in the elements. We were able to implement it using the Map data structure.
We used the Eletism method to select the elements in which we selected the 20 fittest elements from the data set and children that formed after the crossover and mutation.

Our mutation function worked after a fixed amount of iterations and selected two nodes randomly and swapped them. If the mutated route has better fitness than the non-mutated route, then the mutated route is used in place of the non-mutated/original route.

We modified the mutation function to swap random slices, but the time complexity increased, and the results were also not very satisfying.

The GAs proved efficient when quickly bringing down the distance, but after a certain point, the mutations and crossovers could not reduce the distance further.

The table below shows a few of the test runs of the GA algorithm on the **euc_100** dataset.

Number Of Iter.	Mutation Freq	DataSet	Optimal Path
1000	Per 25 moves	euc_100	5790.3132
1000	Per 12 moves	euc_100	4577.5605
5000	Per 25 moves	euc_100	3852.1640

Ant Colony Optimization:

For us, ACO proved to be much better than the GAs, it was faster than the GAs in bringing down the route size but at the same time using more space compared to the GAs.

We placed a minimum of 100 ants and a maximum of 250 ants if the number of cities exceeded 250. After this, we placed them on random points based on the probability matrix and Tabu List to avoid ants moving on the same route.

We also updated the probability of each point being selected every time an ant was placed using the classical Tabu Search algorithm technique.

At each iteration, the ants made a route of which the fittest were stored. Of all the fittest routes the one which was the fittest was printed after some iterations.
After which, we updated the pheromone levels on each route to make sure that the ant follows the optimal path.
The parameters were different for both the **euclidean** and **non-euclidean** distance. And are shown below.

In this we varied the parameters and tried to find the optimal solution of the TSP problem. We used many combinations of all the three parameters and at last settled on the following:

Hyper-Parameters	Euclidean	Non-Euclidean
alpha	2	5
beta	5	5
rho	0.05	0.1

We also tried a bunch of other methods namely:

1) **PSO(Particle Swarm Optimization):** The initial exploration was amazing but it gets stuck on the local minima very soon as compared to ACO so its usefulness was questionable.
2) **Simulated Annealing:** It was used to further optimize the result output by the *PSO* but given the time constraints its performance was very underwhelming.
3) **2-opt and 3-opt heuristics:** The results provided by 3-opt were falling short even when compared to SA and PSO. These algorithms work fine and converge faster than SA but for large datasets(num_cities > 30) it fails to provide satisfactory results in time constrained situations.

This Submission was made by:

- Shubh Agarwal (210020047)
- Saksham Chhimwaal (210010046)