Assignment 6

Pallakonda Sahithi - EE22B080

The python code `ee22b080_tsp.py` implements a solution to the Traveling Salesman Problem using the Simulated Annealing Algorithm.

How to run the code

- Input file should be given as an argument in the command line.
- Input file should be placed in the directory `ee22b080`.
- Open the terminal window in `ee22b080` directory and run the command line given below to get the output.

–> python3 ee22b080_tsp.py ≪InputFile≫

Tasks performed by the Code

It imports the necessary libraries such as `sys` to read the input file, `numpy` for numerical operations and random number generation, `matplotlib` for visualizing data.

Distance Calculation Function:

calculate_distance(city1,city2):

- This function computes the Euclidean distance between two cities represented as (x,y).

distance(cities,cityorder):

- This function calculates the total distance traveled for a given order of cities. It iterates through the ordered cities and accumulates the distances between them.

Algorithm using Simulated Annealing:

'tsp(cities)' Function:

- It initializes the current order of cities and computes the initial total distance.
- It maintains the best order and best distance found throughout the algorithm.
- The parameter(temp) which is analogues to temperature, is set to 100 initially, it decays with a rate of 0.99 for each iteration.

Simulated Annealing Loop: Here's what happens in each iteration:

- A new order is generated by reversing a random segment of the current order.
- The total distance for the new order is computed.
- The change in distance(D) between the new and current order is calculated.

- The code decides whether to accept the new order or not based on a probability distribution. If the new order results in a shorter distance or if it satisfies a probability condition based on the change in distance and the current `temp`, the new order is accepted. If new order is acceptable, then the current order and distance get updated.
- The best order and best distance are updated if the current order is better than the best found so far.
- The `temp` reduces by 0.99 in each iteration, this process continues until `temp` reaches 5×10^{-9}.
- Finally, it returns the best possible order of cities.

Plotting

First, it generates a plot for the random order of cities, then it generates the plot for the shortest possible path found by using simulated annealing algorithm.

Percentage Improvement

Let the length of the randomly generated path is represented as **X**, length of the shortest possible path is represented as **Y**.

Percentage Improvement $= \left(1 - \frac{Y}{X}\right) \times 100$.

Observations

The Best Shortest Possible Distance for the given input file `tsp40.txt` after 15 repetitions is found to be 5.889180666369945, the city order for this path is [8, 35, 2, 7, 32, 4, 12, 5, 29, 0, 9, 28, 37, 39, 25, 23, 15, 14, 11, 3, 22, 10, 38, 34, 17, 19, 20, 30, 33, 13, 24, 18, 16, 1, 21, 26, 6, 36, 31, 27].

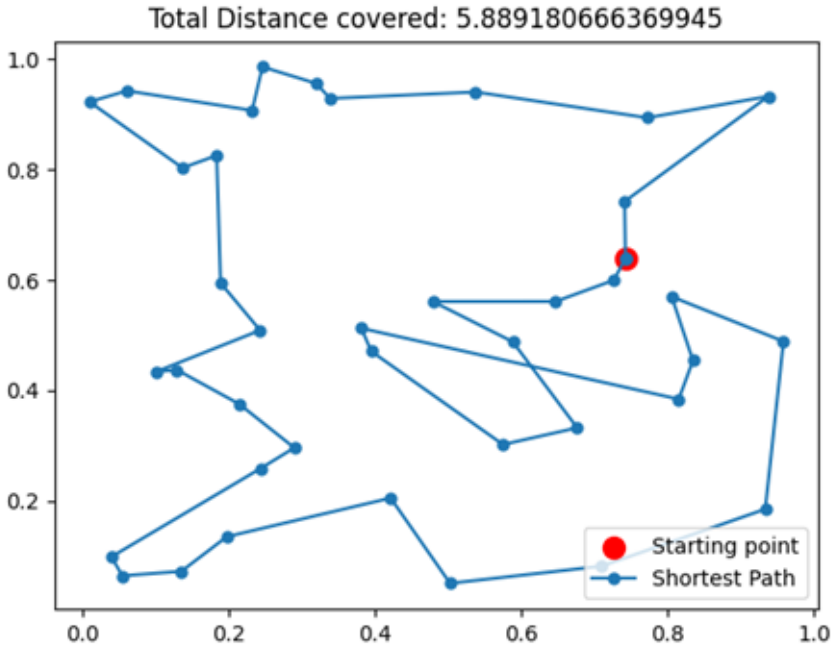

Figure 1: Shortest Path