OPEN ENDED LAB

Travelling Salesman Problem

Submitted by:

ROLL NUMBER	STUDENT NAME
CS-22012	Ramna Kabeer
CS-22017	Hafsa Sohail

Sec-A, Batch-2022

BACHELORS OF ENGINEERING

IN

COMPUTER AND INFORMATION SYSTEMS ENGINEERING

(SEMESTER-V)

UNDER THE GUIDANCE OF MS.HAMEEZA

November 2024

TABLE OF CONTENTS:

1. INTRODUCTION:

The Traveling Salesperson Problem (TSP) is a well-known combinatorial optimization problem where the goal is to find the shortest possible route that visits a set of cities exactly once and returns to the starting city. Due to its NP-hard nature, heuristic and metaheuristic approaches like Genetic Algorithms (GAs) are often used to find approximate solutions efficiently.

This report outlines the implementation and results of solving TSP using a Genetic Algorithm.

2. GENETIC ALGORITHM OVERVIEW:

A Genetic Algorithm (GA) is a search heuristic inspired by natural selection. It uses evolutionary principles such as selection, crossover, mutation, and survival of the fittest to evolve solutions to optimization problems.

3. IMPLEMENTATION DETAILS:

3.1 Parameters:

Parameter	Value
Population size	50
Mutation size	0.01
Number of cities	20
Generations	200

3.2 Steps of the Algorithm

1. **Initialization**:
    a. A population of random routes (permutations of city indices) is generated.
    b. Cities are represented as coordinates in a 2D space.
2. **Fitness Evaluation**:
    a. The **fitness** of a route is calculated as the reciprocal of its total distance:
    Fitness = 1 / Route Distance
3. **Selection**:
    a. Fitness Proportionate Selection (Roulette Wheel) is used to select parent routes. Routes with higher fitness (shorter distances) are more likely to be chosen.
4. **Crossover**:
    a. **Partially Mapped Crossover (PMX)-like approach**:

        i. A random segment of cities is inherited from one parent.
        ii. Remaining cities are added in the order they appear in the other parent, skipping duplicates.
5. **Mutation**:
    a. With a small probability (mutation rate), two cities in the route are randomly swapped to introduce diversity.
6. **Survival and Replacement**:
    a. A new population is formed by combining the selected parents and their offspring.
7. **Termination**:
    a. The algorithm runs for a predefined number of generations, recording the best solution at each step.

4. RESULT:

4.1 Best Route

The algorithm finds a near-optimal route through the cities, which is visualized and discussed below.

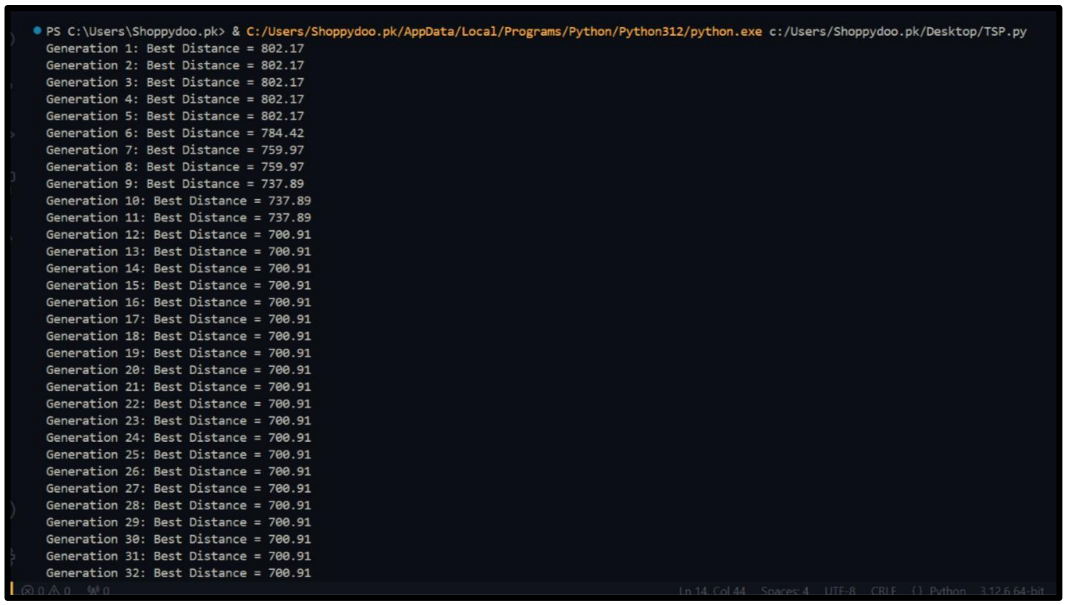
4.2 Performance Metrics

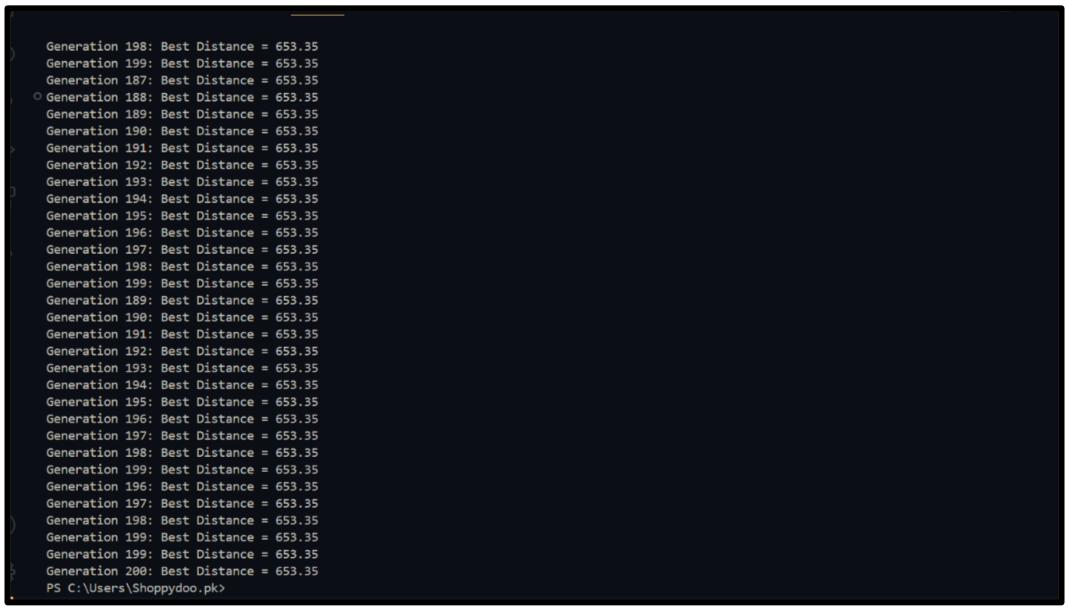
- **Initial Distance**: Distance of the best route in the initial random population.
- **Final Distance**: Distance of the best route after the evolutionary process.

5. VISUALIZATIONS:

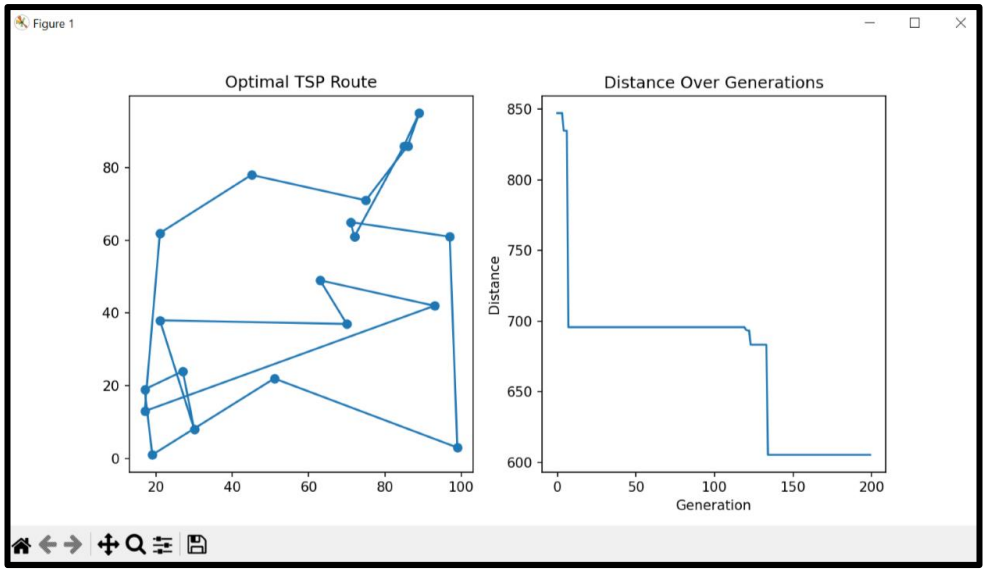
1. **Optimal TSP Route**:
    a. A plot of the best route showing the order in which the cities are visited.
2. **Distance Progression**:
    a. A graph displaying how the shortest distance improves over generations.

6. OUTPUT SNAPSHOTS:


PS C:\Users\Shoppydoo.pk> & C:/Users/Shoppydoo.pk/AppData/Local/Programs/Python/Python312/python.exe c:/Users/Shoppydoo.pk/Desktop/TSP.py
Generation 1: Best Distance = 802.17
Generation 2: Best Distance = 802.17
Generation 3: Best Distance = 802.17
Generation 4: Best Distance = 802.17
Generation 5: Best Distance = 802.17
Generation 6: Best Distance = 784.42
Generation 7: Best Distance = 759.97
Generation 8: Best Distance = 759.97
Generation 9: Best Distance = 737.89
Generation 10: Best Distance = 737.89
Generation 11: Best Distance = 737.89
Generation 12: Best Distance = 700.91
Generation 13: Best Distance = 700.91
Generation 14: Best Distance = 700.91
Generation 15: Best Distance = 700.91
Generation 16: Best Distance = 700.91
Generation 17: Best Distance = 700.91
Generation 18: Best Distance = 700.91
Generation 19: Best Distance = 700.91
Generation 20: Best Distance = 700.91
Generation 21: Best Distance = 700.91
Generation 22: Best Distance = 700.91
Generation 23: Best Distance = 700.91
Generation 24: Best Distance = 700.91
Generation 25: Best Distance = 700.91
Generation 26: Best Distance = 700.91
Generation 27: Best Distance = 700.91
Generation 28: Best Distance = 700.91
Generation 29: Best Distance = 700.91
Generation 30: Best Distance = 700.91
Generation 31: Best Distance = 700.91
Generation 32: Best Distance = 700.91


Generation 198: Best Distance = 653.35
Generation 199: Best Distance = 653.35
Generation 187: Best Distance = 653.35
Generation 188: Best Distance = 653.35
Generation 189: Best Distance = 653.35
Generation 190: Best Distance = 653.35
Generation 191: Best Distance = 653.35
Generation 192: Best Distance = 653.35
Generation 193: Best Distance = 653.35
Generation 194: Best Distance = 653.35
Generation 195: Best Distance = 653.35
Generation 196: Best Distance = 653.35
Generation 197: Best Distance = 653.35
Generation 198: Best Distance = 653.35
Generation 199: Best Distance = 653.35
Generation 189: Best Distance = 653.35
Generation 190: Best Distance = 653.35
Generation 191: Best Distance = 653.35
Generation 192: Best Distance = 653.35
Generation 193: Best Distance = 653.35
Generation 194: Best Distance = 653.35
Generation 195: Best Distance = 653.35
Generation 196: Best Distance = 653.35
Generation 197: Best Distance = 653.35
Generation 198: Best Distance = 653.35
Generation 199: Best Distance = 653.35
Generation 196: Best Distance = 653.35
Generation 197: Best Distance = 653.35
Generation 198: Best Distance = 653.35
Generation 199: Best Distance = 653.35
Generation 199: Best Distance = 653.35
Generation 200: Best Distance = 653.35
PS C:\Users\Shoppydoo.pk>

7. GRAPH:



8. CONCLUSION:

The Genetic Algorithm effectively approximated the solution to the Traveling Salesperson Problem. Key findings include:

- Significant improvement in route distance over generations.
- Genetic operators (crossover and mutation) played crucial roles in finding better solutions.

This implementation highlights the power of GAs in solving combinatorial optimization problems. Future improvements could include experimenting with adaptive mutation rates, alternative crossover methods, or hybrid approaches for enhanced performance.