Solving the Traveling Salesman Problem with Genetic Algorithm Report

Ata Gürsel
Software Engineering
215060044

I. Introduction

Optimization methods provide powerful tools for solving complex problems. This report aims to delve into the application of a genetic algorithm on the Traveling Salesman Problem (TSP), illustrating how the algorithm operates and successfully tackles the challenges posed by TSP.

II. Traveling Salesman Problem (TSP)

TSP is a combinatorial optimization problem where a salesman must visit a set of cities, each exactly once, finding the shortest possible route. As an NP-hard problem, TSP represents the complexity often encountered in combinatorial optimization.

III. Genetic Algorithm Overview

1. **Population Initialization:**
    - Create an initial population consisting of random city routes.
2. **Fitness Evaluation:**
    - Calculate fitness values for each route based on the total distance traveled.
3. **Crossover:**
    - Use crossover operations between parent routes to generate new offspring routes.
4. **Mutation:**
    - Apply mutations to routes with a certain probability.
5. **Parent Selection:**
    - Select parents based on their fitness values.
6. **Elitism:**
    - Preserve the best routes based on an elitism rate.
7. **Visualization:**
    - Visualize the best route and cities in each iteration.
8. **Stopping Criteria:**
    - Terminate the algorithm when a certain iteration limit or fitness level is reached.

IV. Code Structure and Functions

My Python code encompasses fundamental functions for a genetic algorithm applied to TSP. It includes processes such as city coordinate representation, initial population creation, crossover, mutation, parent selection, elitism, and visualization.

V. Parameters and Experiments

Key parameters affecting the performance of the genetic algorithm include population size, iteration count, crossover rate, mutation rate, and elitism rate. Determining these parameters and conducting experiments are crucial for achieving optimal results.

VI. Experimental Results and Analysis

In my experiments, I observed that population size and mutation rate were crucial determinants of performance. Larger populations and lower mutation rates resulted in more diverse routes but extended processing times. Increasing the elitism rate contributed to the algorithm reaching the optimal route more rapidly.

VII. Conclusions and Recommendations

The genetic algorithm proved to be effective in solving TSP. However, further experiments and parameter tuning could enhance its performance. Diversifying crossover and mutation operators could expedite the algorithm's convergence to optimal solutions.

VIII. Visualization and Future Steps

Visualizing the experiment results aided in a better understanding of the algorithm's behavior. For future steps, I plan to conduct experiments using different genetic operators and various parameter settings to further improve the algorithm's overall performance.

IX. Summary

This report comprehensively outlines the application of a genetic algorithm to solve the Traveling Salesman Problem. The obtained results underscore the potential of this optimization method in addressing complex problems.

THANK YOU!