Coding Assignment – Genetic Algorithm Applied to Optimization of Charging Stations

INSTRUCTION: You are reminded to complete this assignment individually and comply with NDSU's statement on Academic honesty, but you can use the basic version of ChatGPT or similar tool ONLY for the coding part of this assignment.

SUBMISSION: Submit your Python code source file as `Assignment_<last-name>.py` and a pdf file as `Assignment_<last-name>.pdf` of your report via Bb.

Overview:

The goal of this assignment is to design an optimization algorithm to select optimal locations for installing electric vehicle charging stations in a city. You will apply your understanding of Genetic Algorithms to solve the problem. Imagine a mid-sized city planning to adopt electric vehicles to reduce carbon emissions and move towards sustainable urban development. To encourage electric vehicle usage, the city needs to strategically place charging stations that maximize accessibility (coverage) for residents and minimize installation and operational costs.
The city government must determine the optimal locations for a fixed number of electric vehicle charging stations (e.g., 20) from a set of candidate locations while balancing several factors:

- Maximizing Coverage: Serve the highest number of residents and electric vehicle drivers within a reasonable distance (e.g., 5 miles).
- Minimizing Costs: Consider budget constraints for installation, maintenance, and operational costs.

Objective:

Demonstrate understanding of the electric vehicle charging station location optimization problem and design the framework for solving it using a genetic algorithm. Implement the genetic algorithm based on your design and apply it to a simplified version of the electric vehicle charging station problem.

Competencies:

- Ability to conceptualize the problem.
- Ability to outline an algorithmic approach for solving the problem.
- Ability to translate problem formulation into code.
- Ability to apply a genetic algorithm to an optimization problem.
- Ability to assess algorithm performance using appropriate metrics.

Tasks:

1. **Formulate the Problem:**
    - Define the key variables (e.g., station locations, demand points, cost of installation).
    - Construct the objective function considering both coverage and cost.
    - List relevant constraints (e.g., budget, distance, grid capacity).

2. **Design a Basic Algorithm:**
    - Describe the outline of a genetic algorithm that could be applied to solve this problem.
    - Include key components: population representation, selection, crossover, mutation, and fitness function.
    - Explain how the GA could balance the objectives (e.g., maximizing coverage while minimizing cost).
3. **Implement the Genetic Algorithm:**
    - Code the genetic algorithm in Python using the basic framework described in Task 2.
    - Ensure the GA includes the following:
        - Representation of potential solutions (station location selection).
        - A fitness function that considers both coverage and cost.
        - Selection, crossover, and mutation mechanisms.
        - A stopping criterion (e.g., number of generations or convergence).
4. **Test the Algorithm:**
    - Apply the GA to a simplified, hypothetical dataset (e.g., 10 candidate locations and 50 demand points).
    - Print the selected charging station locations and their associated costs and coverage.
    - Ensure the algorithm produces reasonable results (i.e., a balance between coverage and cost).
5. **Evaluate Algorithm Performance:**
    - Assess the performance of the algorithm:
        - How well does it cover demand points?
        - What is the total cost of the selected locations?
    - Consider running multiple tests with different parameters (mutation rate, crossover rate, population size).

Deliverables:

- Python code for the genetic algorithm implementation (.py file) including dataset file used (usually in .csv format).
- A report that explains and describes the following (.pdf file):
    - Implemented GA.
    - Test dataset used.
    - Description of approach
    - Results of experiments of different configurations (convergence graph, average fitness, best fitness).
    - Reflections on the strengths and weaknesses of the algorithm.

Evaluation Rubric:

- **Implementation (50%):** Successful implementation of a working solution with testing.
- **Report (50%):** Description of approach and experiments, evaluation of algorithm performance, findings, and proposed improvements.