ACIT 4610

Mid-Term Group Portfolio/Project – 1 -2025

Vehicle Routing Problem Using Quantum-Inspired Evolutionary Algorithms

Objective: Develop and implement a Genetic Algorithm (GA) to solve a basic combinatorial optimization problem in the logistics sector. Specifically, you will address the Vehicle Routing Problem (VRP), a classic problem of finding optimal routes for multiple vehicles to visit a set of locations.

The project is to be carried out in the pre-approved **groups**. Any submission outside this accepted group list will **NOT** be accepted.

Submission Deadline: September 8th (Monday) at 17:00.

Submission Channel: Canvas.

- **Every group must submit Only ONE submission by any of the group members.**
- Note that we will download the coding immediately after the submission deadline to prevent the possibility of updating it later.

Cheating / Plagiarism: In case of any Cheating / plagiarism, it will be handled according to OsloMet's policy. You can find more about it at https://student.oslomet.no/en/cheating

Use of Artificial Intelligence (AI):

- For coding, you can use AI tools as defined by OsloMet's policy.
- For the report, you are NOT allowed to use any AI tools.
- You can find the policy at: https://ansatt.oslomet.no/en/siste-nytt/-/nyhet/veiledning-for-bruk-av-kunstig-intelligens-i-studentoppgaver

1 The Vehicle Routing Problem (VRP)

The Vehicle Routing Problem (VRP) involves finding the most efficient routes for a fleet of vehicles to serve a group of customers from a central depot. The input includes a depot, a set of customers with known locations (and often demands), and a fleet of vehicles with limited capacity. The process must ensure that each customer is visited exactly once, every route begins and ends at the depot, and vehicle capacity or other constraints are not violated. The output is a set of feasible routes, one for each vehicle, that collectively serve all customers. The aim is to minimize the overall cost, usually measured as the total travel distance, travel time, or number of vehicles used.

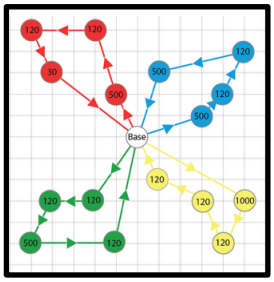

Fig. 1: A VRP

2 Task Details

1 Problem Definition:

a. **Problem Scenario:** Define three groups of scenarios, each with 2 instances. The groups should be categorized as small (a delivery scenario with 1 depot, 2 to 10 vehicles, and 10–20 customer locations), medium (a delivery scenario with 1 depot, 11 to 25 vehicles, and 15–30 customer locations), and large (a delivery scenario with 1 depot, 26 to 50 vehicles, and 20–50 customer locations). Each vehicle starts and ends at the depot and must visit a subset of customers.

b. **Objective:** Clearly state the objective of the VRP. Usually, the aim is to minimize the total distance or travel time for all routes while ensuring each customer is visited exactly once. (For simplicity, you may ignore advanced constraints such as vehicle capacity or time windows unless you decide to include them.).

c. **Example Data:** Provide or describe an example dataset for the VRP. This could include a list of customer locations (with coordinates or a distance matrix between all locations) and the depot location. For example, you might give coordinates for each location and calculate distances as Euclidean distances, or supply a distance matrix for a set of locations. You can choose other formats that suit your implementation. Ensure the example is easy to understand, such as a small table of distances or a simple map of points.

2 **Performance Evaluation Metrics:** Define metric(s) (for example, solution quality, required time, convergence rate, or computational efficiency) to evaluate and compare the performance of GA using our three categories of test problems.

3 **Comparison and Analysis:**

a. Using the above metrics, compare and analyze your implementation using the three categories of test problems.

b. You also need to test and analyse all the test problems for three sets of GA parameters (population size, generation number, crossover probability, and mutation probability).

3 Deliverables

a. **Code Implementation:**

i. Provide a GitHub repository containing all your source code. Include a clear README with instructions on how to install any dependencies and run the algorithms on the provided example data.

ii. Include the GitHub link in the submitted report and ensure it is shareable and executable.

iii. The code should preferably be written in Python (if you are not familiar with Python, you can use a similar language). To handle the VRP data and instances, you may use libraries of your choice, such as NetworkX or OR-Tools.

iv. The repository should be well-organized, and the code should be commented on or accompanied by notes explaining key parts of the GA implementation.

v. The code must be in an executable state and generate the outputs.

b. **Report**

i. You should deliver your report in a PDF format through Canvas.

ii. The project report should be between 1000-1500 words.

iii. The report must mention the Group No as published in the Canvas.

iv. The report should at least discuss the points below:

1. A detailed description of the complete GA design. Describe your implementation in detail so that others can replicate your work. This includes the individual representation, implementation of corresponding genetic operators, selection strategy, termination, and initialization strategies.
2. Define and explain the objective function.
3. Using your specified metric(s), compare and analyze your implementation across our three categories of test problems.
4. A table summarizing the required time to find the solution for all the test problems with your defined three sets of parameters.
5. What are your findings, analysis, and conclusions about the correlation of the parameters based on the achieved results?
6. Describe the effects of these parameter values during the early and later stages of the evolutionary cycle.

NB: Since EAs are heuristic, you must not report results from a single run when presenting the results and comparisons. You should report the best, average, and worst results from several evolutionary runs.