IT 3708, Spring 2020: Project 1
Solving a variant of the Multiple Depots Vehicle Routing Problem (MDVRP) using a Genetic Algorithm (GA)

Lab Goals

- Implement a genetic algorithm (GA) to solve an NP-hard combinatorial optimization problem ─ The Multiple Depots Vehicle Routing Problem (MDVRP).
- Compare the performance of your solutions on several benchmark problems.
- Test and analyze the effects of the genetic operators and related parameters.

Groups Allowed? Groups are allowed, max 2 persons. **Every student** must attend the demo day. The members of a group should sign up for the same time slot on demo day.

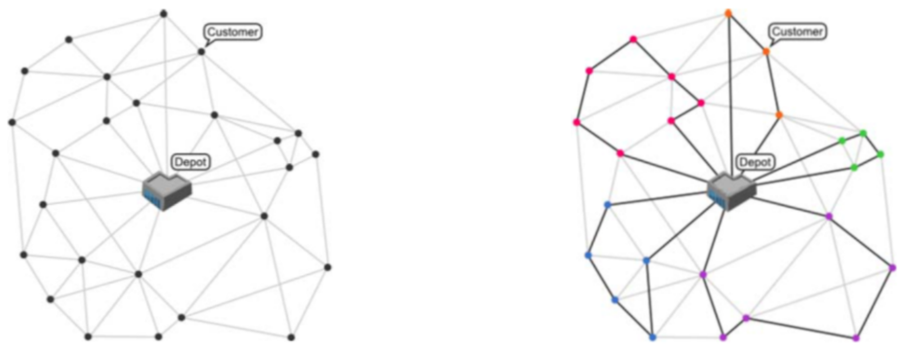
Deadline: February 6, 2020 (Thursday) at 08: 00 AM.

Assignment Details

Vehicle routing problems (VRPs) are classical combinatorial optimization problems which have received much attention in recent years. This is due to their computational complexity, wide applicability, and economic importance. VRP formulations are used to model an extremely broad range of issues in many application fields: transportation, supply chain management, production planning, and telecommunication, to name but a few. In a large number of practical situations and to satisfy real-life scenarios, additional or revised constraints – relative to what is described below – are defined for variants of the VRP.

A typical VRP can be stated as follows:

- A set of geographically dispersed customers with known demands are to be serviced by a (often homogenous) fleet of vehicles with limited capacity.
- Each customer is to be fully serviced exactly once.
- There are a number of vehicles assigned to a depot
- A vehicle starts at a depot and has to return to the same depot.
- The objective is to minimize/maximize some goal, such as, the total distance travelled by all vehicles.



(a) (b)

Fig. 1: A hypothetical instance of a VRP (a) and its solution (b). Note that in this figure, there is only one depot; in general there may be multiple depots. Here, there are five vehicles in the depot.

In this project, you need to solve a variant of the multi-depot VRP (MDVRP):

- There are multiple depots.
- A number of vehicles is assigned to each depot.
- Each vehicle starts at its assigned depot and must return to **the same depot**.
- Every customer should be serviced by a vehicle based at one of several depots.

The MDVRP is NP-hard, which roughly means that an efficient algorithm for solving all problem instances to optimality is unavailable. For this reason, heuristic or meta-heuristic algorithms are good choices to solve the MDVRP. **In this project, you will solve the MDVRP using a well-known bio-inspired algorithm type, namely a genetic algorithm (GA)**.

Problem Formulation: The MDVRP describes the vehicle scheduling challenge for a transportation company. The transportation company has multiple depots from which their vehicles depart and arrive, and has multiple customers being served from the different depots. **The challenge is to make a schedule for each vehicle individually** so that the vehicles drive in the most efficient way optimizing one or several objectives.

Formally, the MDVRP can be defined as follows. We are given a set of depot locations and a set of customer locations, which are assumed to be disjoint (even if two points share the same physical coordinates, they are still handled as different entities). Each customer is characterized by their own demand. A fleet of vehicles with limited capacity is based at each depot. Each vehicle originates from one depot, services the customers assigned to that depot, and returns to the same depot.

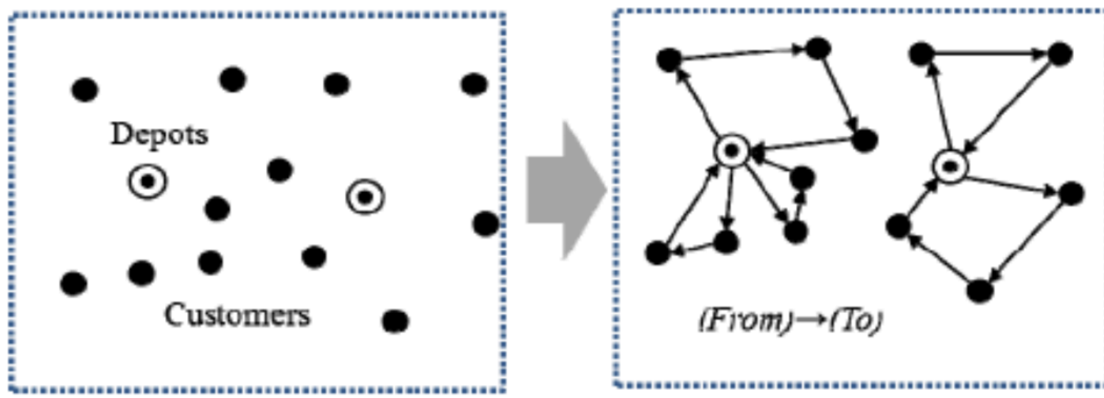
The MDVRP consists of determining the routes for multiple depots with multiple vehicles per depot in parallel. Further, each depot has a set of customers. The route should also optimize predefined objective(s) as well as satisfying the following conditions:

(i) every customer appears on exactly one route.

(ii) every route starts and ends at the same depot.

(iii) capacity limit: the total demand of the customers on any route does not exceed a vehicle's capacity.

(iv) route limit: the total duration of a route does not exceed a preset value *(for this project, it is only for those problems for which this value is mentioned in the test data, that is, the value in the test data is not 0)*.



(a) (b)

Fig. 2: A hypothetical instance of an MDVRP: (a) Locations of customers and depots, customers are shown as black dots while depots are shown as black dots inside circles. (b) One solution, in which three vehicles are assigned to the left depot while two vehicles are assigned to the right depot. The vehicle tours are as indicated with the directed edges (arrows).

Fig. 2 presents an example MDVRP with an arbitrary solution.

Generally, the objectives of the MDVRP are to minimize the total combined route duration for all required vehicles from all depots, to minimize the time spent in serving all customers, or to minimize the number of vehicles needed in serving all customers. In this project, **your main goal is to minimize the total distance travelled by all vehicles across the depots to serve all customers**.

Genetic Algorithm

As mentioned earlier, to solve the MDVRP, you need to implement a genetic algorithm (GA) as presented in the lectures. In order to get optimal or near-optimal results, you may check several forms of representation, genetic operators, and selection mechanisms. It would be beneficial to test whether *elitism* gives a better solution.

Note that GA parameter values (population size, generation number, crossover rate, and mutation rate) are typically correlated and your GA may successfully find optimal or near-optimal solutions if you use the right parameter values. However, there is no definite rule as to how to find such parameter values. Therefore, you should test different sets of parameter values to decide on appropriate values. Study closely the interdependence between the parameter values in your chosen implementation.

(a) Solution for a test problem with 50 customers, each vehicle has a maximum load of 80 and no requirements on maximum duration.

(b) Solution for a larger test problem with 160 customers, each vehicle has a maximum load of 60 and no requirements on maximum duration.

Fig. 3: Solution representations in which depots are illustrated as black squares. While these problems are somewhat different, there are four depots in both (a) and (b).

Things To Do

The 15 points total for this project is 15 of the 100 points available for this course. The 15 points will be distributed in two parts: (i) code demo (12 points) and (ii) questions during demo (3 points).

To test your code, we uploaded several benchmark test data and their solutions. The description of problem and solution data file formats is also included. For the demo session, **your code must have the option to read test data according to the given format**. Note that, your implementation must produce the solution strictly following the graphical (Fig. 3) and text formats (Fig. 4, also mentioned in the provided solution data files). **Each depot should have a unique color for its routes,** as shown in Fig. 3.

When validating how good a solution's **total travel distance** is, refer to the **benchmark table** supplied in the test data. The benchmarks are distances that should be obtainable, while the solution files are the best optimal values found; they can be hard to achieve.

<u>It is not recommended implementing this algorithm in Python. Python generally runs too slow for successful demonstration during the demo day compared to other languages, e.g. Java.</u>

<u>(a) Code Demo, in groups (12p):</u>

There will be a demo session where you will show us the running code and we will verify that it works. In the demo session, you need to describe how you designed and implemented your GA. You have to test you code by running 3 test problems that you will be supplied during the demo. In the demo, you have **<u>30 minutes</u>** to both finish the test problems and answer the questions described in (b). Thus, **<u>the run time of the code is important.</u>**

```
576.87
1  1  60.06  71  1  44 45 33 15 37 17
1  2  66.55  79  1  42 19 40 41 13
1  3  47.00  78  2  25 18 4
2  1  53.44  73  2  6 27 1 32 11 46
2  2  79.47  80  3  48 8 26 31 28 22
2  3  81.40  77  2  23 7 43 24 14
2  4  23.50  54  4  12 47
3  1  50.41  75  3  9 34 30 39 10
```

3	2	25.22	54	4	49 5 38
4	1	47.67	67	4	35 36 3 20
4	2	42.14	69	4	21 50 16 2 29

Fig. 4: Expected text format for your solution

The point distribution for the demo is as follows: Testing of 3 problem instances (12 p = 4 p x 3).

Does your solution find the benchmark value for travel-distance for all vehicles across the depots in serving all customers? (4 p)

* If your value is within 5% of the benchmark value, or better than the benchmark value, you will get 4 points (full score).
* If your value is within 10% of the benchmark value, you will get 3 points.
* If your value is within 20% of the benchmark value, you will get 2 points.
* If your value is within 30% of the benchmark value, you will get 1,5 points.
* Otherwise, you will get 0 points.

You can only get a maximum of 1 point per test if your output is not in the correct format as described (graphically as well as in text format).

(b) Questions during demo, individually or in groups (3p):

During the demo you will be asked questions regarding the behavior and implementation of your code. The exact questions will not be given in advance, but typical questions could be:

- Make changes to the code to see what effect(s) the change has on the score (the total distance travelled by all vehicles). If there is an improvement in the score, you need to be ready to explain the reason behind this improvement. If there is no improvement, why not?
- Describe and show different parts of the code, such as the chromosome representation, mutation operation(s), crossover operation(s), etc.

Any questions regarding this assignment will only be handled at the lab hours or on the Blackboard forum.

Delivery Method and Deadline

You should deliver a zip file with your code on *BlackBoard*. The submission system will be closed at **08:00 AM on February 6, 2020**.

The demo day for Project 1 is also February 6. A signup schedule will be announced one week before. Please follow Blackboard for details.

Every student must submit their (jointly) developed code. You must attend the demo individually on the scheduled demo date. No early or late submission or demo will be entertained except in an emergency.