IT3708 Spring 2021: Project 2

Solving a variant of the Multi-Depot Vehicle Routing Problem (MDVRP) using a Genetic Algorithm (GA)

Lab Goals

- Implement a GA to solve an NP-hard combinatorial optimization problem - The Multi-Depot 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? 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.

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

Deadline: Monday March 22nd 2021

Assignment details

Vehicle routing problems (VRPs) are classical combinatorial optimization problems which have received much attention in recent years [3,4,5]. 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 [3,4,5]:

- 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.
- Several vehicles are assigned to a depot.
- A vehicle starts at a depot and has to return to the same depot.
- The objective is to minimize or maximize some goal. An example objective is to minimize the total distance travelled by all vehicles.

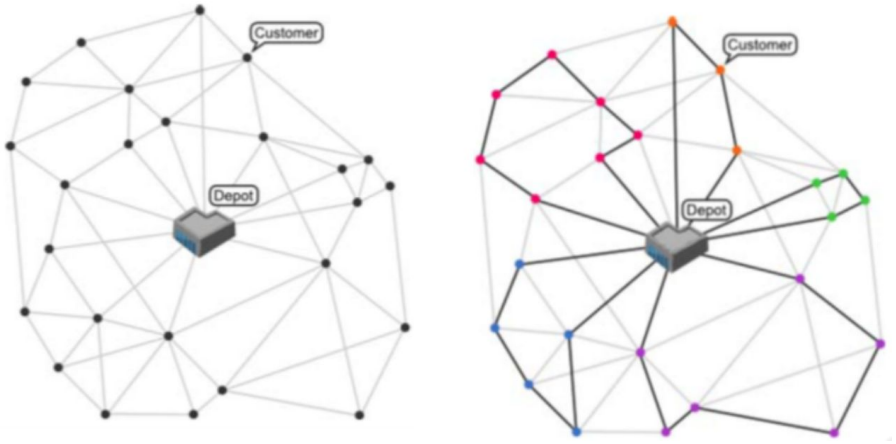


Fig. 1: A hypothetical instance of a VRP (a) and its solution (b). Note that in this figure, there is only one depot; in general there may be multiple depots. Here, there are five vehicles in the depot.

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

- There are multiple depots.
- Several vehicles are assigned to each depot.
- Each vehicle starts at its assigned depot, visits customers and returns to the same depot that it started from.
- Every customer should be serviced by a vehicle based at one depot.

The MDVRP is NP-hard, which roughly means that an efficient (poly-time) algorithm for solving all problem instances to optimality is likely 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) [1,2,3].

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 best possible way, optimizing one or several objectives.

Formally, the MDVRP variant we study is 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. The route should optimize predefined objective(s) as well as satisfying the following conditions:

1. every customer appears on exactly one route.
2. every route starts and ends at the same depot.
3. capacity limit: the total demand of the customers on any route does not exceed a vehicle's capacity.
4. route limit: the total length of each vehicle's route does not exceed a preset value. This requirement is only active for problems that specify a route limit, that is, for problems in which the route limit in the test data is not 0.

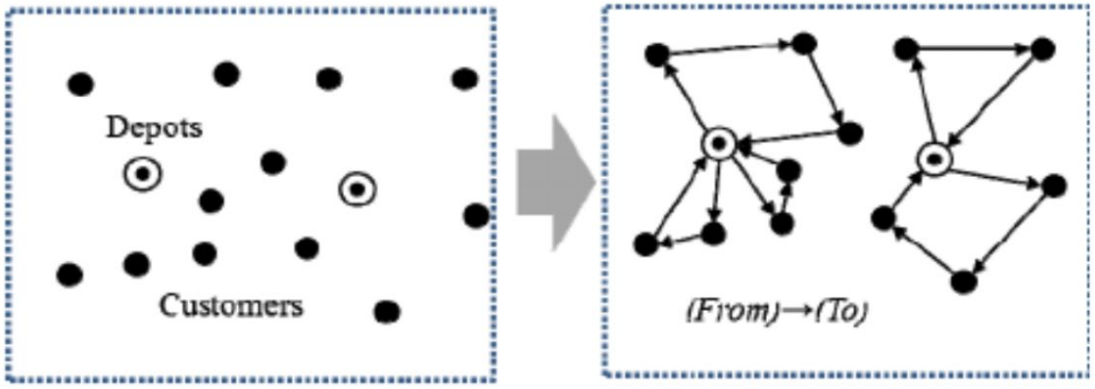


Fig. 2: A hypothetical instance of an MDVRP: (a) Locations of customers and depots, customers are shown as black dots while depots are shown as black dots inside circles. (b) One solution, in which three vehicles are assigned to the left depot while two vehicles are assigned to the right depot. The vehicle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

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

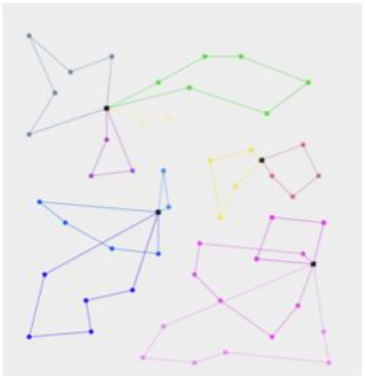
As mentioned earlier, to solve the MDVRP, you need to implement a genetic algorithm (GA) as seen in Project 1. In order to get optimal or near-optimal results, it is recommended to read the article *"Using Genetic Algorithms for Multi-depot Vehicle Routing"* [3] for advice on representation, initialization, and mutation and crossover operators. It would also be beneficial to test whether elitism gives a better solution, and perhaps also consult the other pointers among the references if needed.

Things to Do

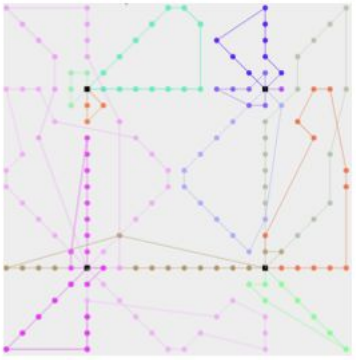
The 20 points total for this project is 20 of the 100 points available for this course. The 20 points will be distributed in two parts: (i) theory and implementation questions during demo (5 points) and (ii) performance-focused code demo (15 points).

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, Julia, C++ etc.

To test your code, we upload several benchmark problem instances and their solutions. The description of problem and solution data file formats are 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 (Figure 3) and text formats (Figure 4 gives an example). Each depot should have a unique color for its routes.



(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).

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.

It is up to you which ***plotting*** library to use. Plotting libraries that have been successfully used for implementing the graphical format include:

- JavaFX (The Canvas class)
- C++ library
- Matplotlib/seaborn/plotly in Python. You would then implement your GA in a faster language, but do the plotting in Python.

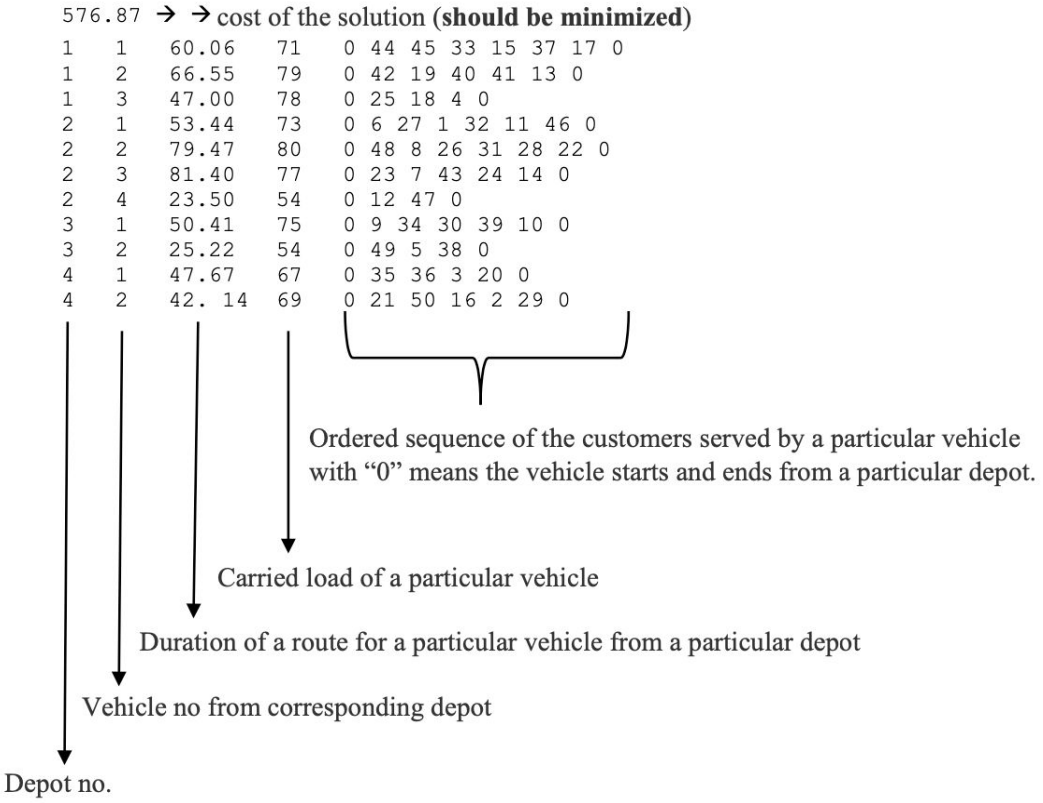


Fig. 4: Expected text format for your solution

Demo (20p)

(a) Questions during demo (5p):

During the demo you will be asked questions regarding the behavior and implementation of your code. In these discussions, you may also be asked to explain certain concepts from the theory of this course. 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), and crossover operation(s). Also be ready to discuss alternatives and the effects of your choices ***for example*** with regards to the exploration-exploitation trade-off.
- Explain related theory concepts from the course's curriculum.

(b) Performance tests of your code (15p):
In this part of the demo you will show us the running code and we will verify that it works. You will be given ***three unseen problem instances*** that you will run your implemented GA on. You have 30 minutes to both finish the test problems (which is the entirety of the demo session). ***Thus, the run time of the code is important***. The questions described in a) will be asked as your code is solving the instances.

The point distribution is as follows: Testing 3 problem instances (15 points = 5 points x 3).For

For each problem instance, you can get full or partial scores as follows:

- If your value is within 5% of the benchmark value, or better than the benchmark value, you will get 5 points (full score).
- If your value is within 10% of the benchmark value, you will get 4 points.
- If your value is within 20% of the benchmark value, you will get 3 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).

Delivery Method and Deadline

You should deliver a zip file with your code on BlackBoard. The submission system will be closed on ***Monday March 22nd at 08:00 AM***.

The demo day for this project is also ***Monday March 22nd***. 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.

References

[1] A. E. Eiben and J. E. Smith. “Introduction to Evolutionary Computing,” 2nd Edition, Springer 2015, pages 67 - 70 (permutation representation) & pages 203 - 211 (constraint handling).
[2] D. Simon. “Evolutionary Optimization Algorithms,” Wiley 2013, pages 449 - 478 (combinatorial optimization incl. TSP).
[3] B. Ombuki-Berman and F. T. Hanshar. “Using Genetic Algorithms for Multi-depot Vehicle Routing.” In: F. B. Pereira and J. Tavares (eds). “Bio-inspired Algorithms for the Vehicle Routing Problem.” Studies in Computational Intelligence, vol 161. Springer 2009. https://doi.org/10.1007/978-3-540-85152-3_4
[4] J. R. Montoya-Torres, J. L. Franco, S. N. Isaza, H. F. Jiménez, N. Herazo-Padilla. “A literature review on the vehicle routing problem with multiple depots.” Computers & Industrial Engineering, Volume 79, 2015, pages 115-129, https://doi.org/10.1016/j.cie.2014.10.029.
[5] K. Braekers, K. Ramaekers, I. V. Nieuwenhuyse. “The vehicle routing problem: State of the art classification and review.” Computers & Industrial Engineering, Volume 99, 2016, pages 300-313, https://doi.org/10.1016/j.cie.2015.12.007.