VRP Challenge (v2)

Difficulty level: advanced

Keywords

- Mathematical modelling
- Mixed Integer Linear Programming
- Excel Solver
- Python+PuLP
- Heuristic algorithms
- Exact algorithms

Introduction

The supermarket chain *SuperAmazingMarket* has decided to launch a new service: home delivery of groceries.

Each customer who requests the service, located in the province of Brescia, must be given a certain number of shopping bags containing the requested goods.

Each supermarket participating in the service, also located within the province, have some small, medium, and large capacity vehicles available for delivery: the small capacity vehicle can carry at most 30 shopping bags, the medium vehicle can carry at most 50, while the large one at most 70.

Each day, each vehicle will load the goods from the supermarket in which it is located, and set off to make deliveries. Each vehicle can stay on the road for a maximum of 3 hours, from departure until it returns to the same supermarket from which it started.

During the journey, each vehicle can pass (even more than once) next to customers, without necessarily serving them, and next to the supermarkets, including its own (but without stocking up on other bags).

Once back at its supermarket, the vehicle will no longer be able to leave. It is expected that each courier (who will drive the vehicle and deliver the shopping) will cost the company 50 €/day. Also, it will take about 1 minute to bring a pair of bags from the vehicle to the customer's home. During the actual delivery phase, the vehicle will be shut down. Finally, it has been estimated that the fuel for the vehicles will cost 0.07 €/minute.

The company *SuperAmazingMarket* asks for your help in trying to spend as little as possible, by establishing:

- which vehicles from each supermarket will make the deliveries;
- which path each of these vehicles must follow;

- which customers each of these vehicles should serve.

In particular, the company is interested in solving the problem on three hypothetical different scenarios, characterized by:

- a different number of customers to serve;
- a different number of supermarkets;
- a different number of vehicles available and their allocation to the various supermarkets.

In each scenario, it is also provided how many shopping bags each customer must receive and which roads the vehicles can travel.

All this information is summarized in three different text files, one for each scenario, attached to this document (i.e., the files `instance-demo.txt`, `instance-1.txt`, and `instance-2.txt`). Hereafter follows an example instance:

```
CLIENTI: 20          Customers
SUPERMERCATI: 2      Supermarkets
VEICOLI: 4           Vehicles

ID-DOMANDA :  ID-DEMAND (related to supermarkets and customers)
1 0 // supermercato 1
2 0 // supermercato 2
3 9 // cliente con ID 3 che richiede 9 borse della spesa
4 2 // cliente con ID 4 che richiede 2 borse della spesa
5 8 // cliente con ID 5 che richiede 8 borse della spesa
...

NODO1-NODO2-TEMPO :  NODE1-NODE2-TIME (travelling times)
1 3 20 //per percorrere il tratto stradale che collega il punto 1 al punto 3 servono 20 minuti
1 4 26 //per percorrere il tratto stradale che collega il punto 1 al punto 3 servono 26 minuti
1 10 38
1 11 28
2 8 71
3 6 18
3 9 37
...

ID-CAPACITA-SUPERMERCATO :  ID-CAPACITY-SUPERMARKET (related to vehicles)
1 30 1 //il veicolo 1 ha una capacità di trasporto di 30 borse ed è assegnato al supermercato 1
2 50 1 //il veicolo 2 ha una capacità di trasporto di 50 borse ed è assegnato al supermercato 1
3 70 2 //il veicolo 3 ha una capacità di trasporto di 70 borse ed è assegnato al supermercato 2
...
```

The first rows of the "ID-DEMAND" section always refer to the supermarkets, characterized by a demand equal to 0. The following rows assign a progressive number to each customer and indicate how many shopping bags each customer asks for.

Arcs not appearing in the "NODE1-NODE2-TIME" section CANNOT be travelled.

The vehicle with ID 3 has a capacity of 70 shopping bags and it is assigned to the supermarket with ID 2

Tasks

In the file `vrp_challenge.py`, you will find the implementation of the mathematical model of the VRP Challenge problem. In particular, there are:

- the `read_input` function reads an instance from a text file;
- the decision variables of the problem;
- **some** constraints.

Indeed, unfortunately, the problem objective function and other constraints were accidentally deleted. Fortunately, anyway, some suggestions and comments describing what those constraints imposed were left.

1. Read the `vrp_challenge.py` script carefully and understand why certain variables have been defined and some constraints have been imposed.

2. Complete the implementation of the mathematical model in Python by following the suggestions in the comments and solve it by exploiting the PuLP library.