FIRST ASSIGNMENT

March 17, 2020

Decision Support Methods
(CC3003)

Francisco Gonçalves
201604505

Departamento de Ciência de Computadores
Faculdade de Ciências de Universide do Porto

0.1 Introduction

In order to answer the given questions in this first assignment, a mathematical optimization model was made using GLPK. Its implementation is somewhat similar for the two exercises that require it.

Both use the same *.dat* file which specifies all the data needed in order to solve these exercises. Given the number of locations used, resulting in quite a lengthy file the following transcript is merely demonstrative:

Listing 1: Data Specification (assignment.dat)

```
set LOCATION :=
Lisbon
Porto
(...)
Sao_Miguel_do_Couto
Galegos;

param:               population latitude   longitude :=
Lisbon               517802     38.71667   -9.13333
Porto                249633     41.14961   -8.61099
(...)
Sao_Miguel_do_Couto  5416       41.33167   -8.46185
Galegos              5404       41.56268   -8.57204 ;

param radius := 6371.009;

end;
```

The rest of the implementation lies in a *.mod* file and can be changed depending on the decision variables and what this mathematical optimization model needs to achieve which is:

- minimizing sum of Manhattan Distances of all locations to the Distribution Center
- minimizing sum of delivery costs to all locations

0.2 First Exercise

0.2.1 Minimization of sum of Manhattan Distances

The program for the first question is as follows:

Listing 2: Implementation (assignment1.mod)

```
set LOCATION;

param latitude {LOCATION};
param longitude {LOCATION};
param population {LOCATION} > 0;
param radius > 0;

// DECISION VARIABLES
var Xdc; # Latitude of DC
var Ydc; # Longitude of DC

// RESTRICTIONS
var lat{x in LOCATION}; // Absolute difference between latitude of DC and location
s.t. AbsLatDC{x in LOCATION}: lat[x] >= Xdc - latitude[x];
s.t. AbsLatX{x in LOCATION}: lat[x] >= latitude[x] - Xdc;
s.t. AbsLat{x in LOCATION}: lat[x] >= 0;

var lon{x in LOCATION}; // Absolute difference between longitude of DC and location
s.t. AbsLonDC{x in LOCATION}: lon[x] >= Ydc - longitude[x];
s.t. AbsLonX{x in LOCATION}: lon[x] >= longitude[x] - Ydc;
s.t. AbsLon{x in LOCATION}: lon[x] >= 0;

s.t. LimitRadius: sum {x in LOCATION} (lat[x] + lon[x]) <= radius;

var tot{x in LOCATION}; // Sum of latitude and longitude between DC and location
s.t. TotDist{x in LOCATION}: tot[x] = lat[x] + lon[x];

// OBJECTIVE
minimize distance:
    sum {x in LOCATION} (tot[x]); // Or lat[x] + lon[x]

end;
```

The *set LOCATION* as well as the four parameters that follow are used to obtain the data specified in the file previously mentioned.

Afterwards, the two decision variables, Xdc and Ydc, are defined as the latitude and longitude of the Distribution Center (DC), respectively. These are required in order to obtain the Manhattan Distance to all the locations already defined.

The next eight lines are needed to ensure that when subtracting the latitude and longitude between the decision variables and one of the values of a location you always obtain a positive result, as some coordinates can be negative.

Finally, the objective function is defined to suit whatever needs to be obtained, which is for the first exercise, the least possible distance to all locations.

In this implementation, given technical difficulties the distance obtained from subtracting the latitudes and longitudes could not be converted to the metric system and be displayed in kilometers. The lowest value that appears in the *.sol* file is still the closest location to the DC.

After running the program, a *.sol* file is created with the requested result. In this case, given the overwhelming amount of variables, the said file is quite extensive, however here are the important sections needed to answer the exercise:

Listing 3: Results (assignment1.sol)

```
Problem:   assignment1
Rows:      2655
Columns:   1139
Non-zeros: 6064
Status:    OPTIMAL
Objective: distance = 603.25433 (MINimum)

(...)

879 tot[Condeixa-a-Nova]  B   0.13792

(...)
```

Therefore, the minimum possible distance from the DC to all the locations is **603.25433**. Although, this result is merely the sum of the differences of latitudes and longitudes between the DC and the locations and not an actual distance in kilometers, therefore it is inaccurate as a conversion to the metric system using the Haversine formula is needed.

The location that is closest to the optimal position of the Distribution Center, according to this mathematical optimization model, is **Condeixa-a-Nova**, as it shows the lowest value when comparing the values of tot for all locations.

The coordinates determined as optimal for the DC can also be inspected under the names of the given decision variables (Xdc and Ydc).

0.3 Second Exercise

0.3.1 Minimization of sum of total delivery costs

Listing 4: Programa (GLPK)

```
set LOCATION;

param latitude {LOCATION};
param longitude {LOCATION};
param population {LOCATION} > 0;
param radius > 0;

// DECISION VARIABLES
var Xdc; # Latitude of DC
var Ydc; # Longitude of DC

// RESTRICTIONS
var lat{x in LOCATION}; // Absolute difference between latitude of DC and location
s.t. AbsLatDC{x in LOCATION}: lat[x] >= Xdc - latitude[x];
s.t. AbsLatX{x in LOCATION}: lat[x] >= latitude[x] - Xdc;
s.t. AbsLat{x in LOCATION}: lat[x] >= 0;

var lon{x in LOCATION}; // Absolute difference between longitude of DC and location
s.t. AbsLonDC{x in LOCATION}: lon[x] >= Ydc - longitude[x];
s.t. AbsLonX{x in LOCATION}: lon[x] >= longitude[x] - Ydc;
s.t. AbsLon{x in LOCATION}: lon[x] >= 0;

s.t. LimitRadius: sum {x in LOCATION} (lat[x] + lon[x]) <= radius;

var tot{x in LOCATION}; // Sum of latitude and longitude between DC and location
s.t. TotDist{x in LOCATION}: tot[x] = lat[x] + lon[x];

var cost{x in LOCATION}; // Delivery cost from DC to location
s.t. TotCost{x in LOCATION}: cost[x] = 3 * (population[x]/1000) * tot[x];

// OBJECTIVE
minimize delivery_cost:
   sum {x in LOCATION} (cost[x]); // Or 3 * (population[x]/1000) * (lat[x] + lon[x])

end;
```

Most of the implementation remains the same as the optimal location for the Distribution Center must be found again.

However, in this second exercise we must consider delivery costs which rely on distance to a given location as well as its number of inhabitants as opposed to just the distance in the previous exercise.

Therefore, the decision variables remain the same as we still need to calculate the existing distances. The same thought process is valid for the restrictions which are related to the distance limits.

Finally, the objective function is modified in order to reflect the sum of, not distances to the DC but its distribution costs and since three deliveries are needed per one thousand of the population of a location the following expression is multiplied by the previous objective function: $3 * (population[x]/1000)$ where x is a given location.

Just like in the previous implementation, the same difficulties in obtaining the distance in kilometers are evident here which obviously reflects on a wrong result as the objective requires a distance in kilometers and not a sum between the subtractions of latitudes and longitudes.

After running the program, a *.sol* file is created with the requested result. In this case, given the overwhelming amount of variables, the said file is quite extensive, however here are the important sections needed to answer the exercise:

Listing 5: Results (assignment2.sol)

```
Problem:   assignment2
Rows:      3034
Columns:   1518
Non-zeros: 6822
Status:    OPTIMAL
Objective: delivery_cost = 34123.87359 (MINimum)

(...)

810 tot[Santarem] B   0.06957

(...)
```

Therefore, the minimum possible delivery costs from the DC to all the locations (assuming the distance is merely the sum between the subtractions of latitudes and longitudes between the optimal DC coordinates and the locations' coordinates) is **34123.87359**. It is, once again, worth noting that just like in the first exercise, this result is also inaccurate for similar reasons.

As for the closest location to the optimal location of the Distribution Center, by inspecting the *.sol* file and inspecting the values exactly like it was done for the previous question, the correct answer is **Santarém**.

The coordinates determined as optimal for the DC can also be inspected under the names of the given decision variables (Xdc and Ydc).

0.4 Third Exercise

In order to answer the third and last exercise, there is no need for a new mathematical optimization model as the answer can be obtained from the previous exercise's implementation by checking the *.sol* file:

Listing 6: Results (assignment2.sol)

```
(...)

1140   cost[Lisbon]  B   1609.69
1141   cost[Porto]   B   1437.17
1142   cost[Amadora] B   588.396
1143   cost[Braga]   B   914.352
```

Therefore, when consulting the *.sol* file already obtained when running the preceding implementation when looking through each location's delivery cost we can easily come to the conclusion that the one with the largest distribution cost is **Lisbon**, which makes sense given its population being noticeably greater than any other location's.