uc3m | Universidad Carlos III de Madrid

Course 2022-2023

Heuristics and Optimization

LINEAR PROGRAMMING LAB

https://github.com/saralozanoserroukh/Heuristic_lab

Sara Lozano Serroukh NIA:100452223

Rodrigo De Lera De Las Heras NIA:100452323

Contents

INTRODUCTION

In this document, the development of a Linear Programming task will be explained.

The assignment we were given consisted of three tasks.

For the first task we had to model a Linear Programming problem and solve it taking advantage of the solver provided by Spreadsheets LibreOffice. The problem consists in generating the optimal routes for some buses to pick up all the students from their bus stops and drive them to the school.

For the second task we needed to optimize a similar problem to the bus route problem described before with the difference that this time, we do not know which students go to which bus stop. This is a new decision that the model to be developed will need to make. For this second task, we are asked to model the problem in MathProg. In addition, we must implement the model in task 1 also in MathProg.

Finally, the third task was to provide an analysis of the obtained results. A complexity analysis is also included, as requested, along with the advantages and drawbacks of using each of the tools (LibreOffice and GLPK).

MODEL DESCRIPTION

Problem part 1

After many models to solve this problem, we arrived at an idea that was crucial for us for starting to design our solution: we can treat this task as a transportation problem where we need to move "empty seats" from the parking lot to the bus stops. Each bus can hold at most 20 "empty seats", and we must reach the school with 0 seats, meaning that every student has been picked up from the stop and has made it to the school.

When modelling this linear programming task, the decision variables, the objective function and the constraints to which the function is subjected to must be defined.

Decision variables:

According to our model, we have two types of decision variables. Firstly, we defined a set of binary variables. This set consists of a 5x5 matrix in which we represent the routes of the buses. For instance, if a bus that leaves from parking reaches the first stop, a 1 should appear in the solution in the second position of the first row. Then, we defined another 5x5 matrix, but in this case, these integer variables will control the flow of students in each bus, which show the number of "empty seats" that are transported from the stop in the rows to the one in the columns

Objective function:

The aim of the task is to minimize the cost associated with the school bus routes. Therefore, the function to minimize will be of the form: 120 * "number of buses" + 5 * "kilometers traveled". The number of buses is obtained with the addition of the matrix's first row of the binary decision variables, which show the number of buses that leave the parking lot; whereas the kilometers traveled will be obtained using the SUMPRODUCT function of Spreadsheet, which will return the addition of the solution of the product of the matrix binary decision variables and the cost matrix.

Constraints:

The objective function is subjected to the following constraints

- The same number of buses that leave the parking lot have to enter the school, which means that the sum of the first row of the matrix of binary variables has to be equal to the sum of the last column of that matrix.

- Each stop has to be visited from 1 bus, which means also that from each stop only 1 bus leaves. The corresponding rows/columns to the stops have to be each one equal to 1.

- To control the flow of students, we know that inflow – outflow = demand. The inflow corresponds to the addition of each column in the flow matrix, whereas the outflow is

computed with the addition of each row of the flow matrix. The demand is the number of students in each stop.

- To relate both types of decision variables, we add a constraint to specify that each element in the flow matrix has to be less or equal than 20 times each element of the binary matrix. This way, we can make sure that we do not surpass the capacity of each bus.

Once the problem has been written and solved by the spreadsheet solver, the obtained result for the objective function was 400€. The optimal school bus route obtained was using two buses, the first bus would pick up the students from the first and then travels to the second stop, and the other bus would pick up the students from the third stop. This can be seen better in the following image:

DECISION VARIABLES (binary)	PARKING	S1	S2	S3	SCHOOL	leaving buses
PARKING	0	1	0	1	0	2
S1	0	0	1	0	0	1
S2	0	0	0	0	1	1
S3	0	0	0	0	1	1
SCHOOL	0	0	0	0	0	0
entering buses	0	1	1	1	2	

This first problem was also implemented in MathProg. In order to create a general problem solver, firstly we created a data file which we named “Model1Data.dat”. In this file we organized all the data we were given. We established a set of STOPS where we included the parking, the 3 stops and the school. Afterwards, some parameters were defined. Those parameters are the cost per bus, the cost per kilometer, the bus capacity, the cost in kilometers to arrive for one stop to another and the demand (the number of students per stop). Later, we created the file “Model1.mod”, in this file we first initialized the sets and parameters described in the data file. Then, we initialized the decision variables as binary and integer and to be non-negative, both decision variables are matrices with size 5x5 as in the Spreadsheet. After that, we wrote the objective function and the constraints to which it is subjected to. The optimal solution found by the solver was:

```
saralozano@DESKTOP-TL3G99F:~/heuristic/lab1$ cat Model1_sol.txt
Problem:    Model1
Rows:       38
Columns:    50 (50 integer, 25 binary)
Non-zeros:  153
Status:     INTEGER OPTIMAL
Objective:  budget = 400 (MINimum)
```

Problem part 2

To model the second part, we used the part one as a basis and change the needed items. In “Model2Data.dat” we included the data for this problem. Beside the set of STOPS, we included a set of STUDENTS, we uploaded the needed parameters values with the new provided ones and added some other new parameters. There new parameters are siblings, which provide the information of which students are related and possible_stops, that provide information about to which stop each student can go. Then, as in “Model1.mod”, in “Model2.mod” we initialized the sets, parameters, decision

variables, objective function and its constraints. To the decision variables, apart from the ones in “Model1.mod”, we included a binary matrix to assign the students to the stops. Moreover, some new constraints were needed to select the correct stop and check if the students are siblings as related students must go to the same stop. The main challenge in this part is that we do not have a fixed demand per stop, since we have a moving group of students, so the constraint involving inflow, outflow and demand needed to be changed. The optimal solution found by the solver was:

```
rodri@DESKTOP-AEHBHRE:~/heuristics/tutorial$ cat model2sol.txt
Problem:    Model2
Rows:       404
Columns:    90 (90 integer, 65 binary)
Non-zeros:  261
Status:     INTEGER OPTIMAL
Objective:  budget = 585 (MINimum)
```

What is more, the solution provided the following bus routes: three buses are leaving the parking lot, each one to each one of the stops. Then, after visiting the corresponding stop, each bus went to the school. We can also see the students' distribution among the stops. In S1 we can find A1, A2, A3, whereas in S2 we find A4 and A5 and in S3, A6, A7 and A8.

ANALYSIS OF RESULTS

PROBLEM 1

The first thing to analyse is the complexity of the problem. To solve this problem, we defined 50 decision variables and 5 constraints.

For this part, it can be easily checked that all the constraints are satisfied.

In the table, we can see represented in the rows the leaving buses and in the columns the arrivals. So, as we can see, there are 2 buses leaving the parking and 2 buses arriving at the school, which satisfies the constraint that the same number of buses that leave the parking must reach the school. Then, let's consider the bus that arrives at the first stop, as we can see in the second row, that bus goes then to the second stop and afterwards to the school. Then, the bus that arrives at the third stop goes later to school. As we can see, in the corresponding rows and columns for the stops, there is only one 1, which implies that each stop is only visited once and from each stop only one bus leaves.

DECISION VARIABLES (binary)	PARKING	S1	S2	S3	SCHOOL	leaving buses
PARKING	0	1	0	1	0	2
S1	0	0	1	0	0	1
S2	0	0	0	0	1	1
S3	0	0	0	0	1	1
SCHOOL	0	0	0	0	0	0
entering buses	0	1	1	1	2	

Then, to control the flow a different matrix was used.

DECISION VARIABLES (flow)	PARKING	S1	S2	S3	SCHOOL	inflow	outflow	in-out	demand
PARKING	0	20	0	10	0	0	30	-30	
S1	0	0	5	0	0	20	5	15	15
S2	0	0	0	0	0	5	0	5	5
S3	0	0	0	0	0	10	0	10	10
SCHOOL	0	0	0	0	0	0	0	0	
inflow	0	20	5	10	0				
demand		15	5	10					

What we can see here is that 20 “empty seats” go from the parking lot to S1, and 10 “empty seats” go from the parking lot to S3. Then, we can observe that 5 “empty seats” go from S1 to S2, which means that we have transported 20-5=15 seats for the 15 students in S1. Finally, we cannot see the path between S2-school and S3-school because we are moving 0 “empty seats” from both stops to the school because we have given the “empty seats” to the students in S2 and S3.

Later, we created some new problems, changing the provided data to check the functionality and complexity of the given problem. These new problems were executed with MathProg.

- If the distance between S1 and S2 is incremented to 15, we obtain:

```
saralozano@DESKTOP-TL3G99F:~/heuristic/lab1$ cat sol_model1_modi.txt
Problem:    Model1
Rows:       38
Columns:    50 (50 integer, 25 binary)
Non-zeros:  153
Status:     INTEGER OPTIMAL
Objective:  budget = 405 (MINimum)
```

If we look in the text file, we can see that the new routes are: one bus leaves from parking to S1 and another one to S2, the bus that goes to S1 then goes to school and the bus that goes to S2 then goes to S3 and then to the school.

- If the number of students in S1 is decreased to 5, we obtain:

```
saralozano@DESKTOP-TL3G99F:~/heuristic/lab1$ cat sol_model1_modi2.txt
Problem:    Model1
Rows:       38
Columns:    50 (50 integer, 25 binary)
Non-zeros:  153
Status:     INTEGER OPTIMAL
Objective:  budget = 255 (MINimum)
```

If we look in the text file, we can see that now only one bus leaves the parking lot. The route of this bus is PARKING-S1-S3-S2-SCHOOL. This makes sense since all the students now fit in one bus and the cost per kilometer is less than the cost per bus.

PROBLEM 2

The first thing to analyze is the complexity of the problem. To solve this problem, we defined 90 decision variables and 8 constraints. The variables were distributed in a binary 5x5 matrix for the routes, a 5x5 integer matrix for the flow and a 8x3 matrix for assigning students to each stop. Our five first constraints are the same as in Part 1, but we do not have a demand vector, the demand is set by assigning students to each stop. We also have three constraints that serve the purpose of assuring that siblings go to the same stop, that students only go to the stops reserved to them and that every student goes to a stop.

Later, we created some new problems, changing the provided data to check the functionality and complexity of the given problem.

- If there were no siblings, we obtained:

```
rodri@DESKTOP-AEHBHRE:~/heuristics/tutorial$ cat model2sol.txt
Problem:    Model2
Rows:       404
Columns:    90 (90 integer, 65 binary)
Non-zeros:  241
Status:     INTEGER OPTIMAL
Objective:  budget = 380 (MINimum)
```

If there are no siblings, A4 and A5 will not need to go to the same bus stop. Then, each of them could go to a different bus stop and subsequently only fill two stops: S1 and S3. The new routes for this problem would be: two buses leave the parking lot, one of them goes to S1 and the other to S3 and then both go

to the school. In this case A1, A2, A3 and A4 are in S1, while A5, A6, A7 and A8 are in S3.

- If A3 could go to S1 and S2, we obtain:

```
saralozano@DESKTOP-TL3G99F:~/heuristic/lab1$ cat sol_model2_modi.txt
Problem:    Model2
Rows:       404
Columns:    90 (90 integer, 65 binary)
Non-zeros:  261
Status:     INTEGER OPTIMAL
Objective:  budget = 405 (MINimum)
```

In this case, we would have 2 bus routes, one leaving the parking lot to go to S1 and another to go to S2, the one going to S1 would go then to SCHOOL and the other would go to S3 and then SCHOOL. For this example, A2, A3, A4 and A5 would go to S1, A1 would go to S2 and A6, A7 and A8 would be in S3.

PROS AND CONS

In the statement, we were also asked to compare the two used tools.

Personally, we consider that LibreOffice provides a much cleaner solution, this is due to the fact that we organized the data and how to present the output result; whereas, in GLPK the result is provided in a text file.

The solution provided by GLPK can be clearly seen and it also includes some extra information that SpreadSheets will not show, but when having a big amount of variables, this file may be a little overwhelming.

However, we believe that GLPK provides a simpler implementation method. Writing the constraints in MathProg seemed to us easier than working with LibreOffice; what is more, writing the decision variables and the data sets was less tedious.

CONCLUSIONS

Our main conclusion drawn from the project is that the most difficult task in this area of study is creating a model of our problem, implementing it can be easily done. In our case, we needed approximately 20 hours to model the problem and implement it in LibreOffice, whereas we only needed to use 5 hours to implement both tasks in GLPK, apart from testing different datasets and situations to check if our model worked for the widest set of similar problems only changing our data file.

Another deduction is that the use of computers is mandatory to obtain solutions for optimization problems, since calculating a Simplex algorithm by hand for 5 variables, for example, is an arduous task, and in this project, we used 50 variables for Task 1 and 90 variables for Task 2. In our opinion, solving that system of inequalities by hand is impossible.

Finally, we have also concluded that most complex optimization problems can be modelled as a combination of more general models, for example, we used a transportation model for Task 1, and a combination of transportation and assignment for Task 2.