Decision-making in Transport and Logistics (1CM110)

ASSIGNMENT 1

Deadline: Nov 30th, 13:00

Dr. Layla Martin, Dr. Rolf van Lieshout, Dawei Chen

1. READ the **entire** assignment before you start. Get an idea of what you need to do and assess how much time you will need.
2. Do not split the work within you group alongside tasks or questions (e.g., person A works only on the report, person B only on the code, or person A works on question 1, and person B on question 2), but collaborate on all questions and all tasks.
3. It is **not** allowed to collaborate with people outside your group. If we observe this, all involved groups will be fined.
4. Copying code or text from others is considered plagiarism.
5. Test your implementations on small instances you constructed yourself, and verify that the results are correct. It is often good practice to write a little bit of verification code to verify the correctness of your solutions (don't just rely on your models).
6. If you solve your model on a (more or less) recent computer, and it is taking more than 30 minutes, something is wrong with your model.
7. Whenever you are using Gurobi, do NOT disable its output: you want to be able to see what's going on when the solver is solving your models.
8. All your Gurobi models must be linear models. Note that Gurobi does not throw an error if your model is non-linear, so you must check this yourself!
9. Code must be well-documented and easily readable. We will deduct points if we do not understand what the code does or cannot replicate your results.
   Tip: Use separate functions for managing the tests (e.g., calling other functions), input handling, output handling, creating math models, and solving math models.

10. This assignment is deliberately containing some weak specifications. You can use the Canvas discussions as well as the Q&A sessions to obtain clarity.

Goals of this assignment:

1. Enhance your understanding of the course material through hands-on experience.
2. Increase your experience with Python, and make you familiar with the implementation of small models in a commercial solver. Understand and interpret the Gurobi output.
3. Practice eliciting information from weakly specified problems.
4. Presenting actionable insights from sensitivity analyses.

Introduction

FreshMilk supplies supermarkets throughout the Netherlands with dairy products. They collect raw milk from the farmers, ship them to collection sites, make multiple products (milk, yogurt, cream), and ship the products to supermarkets.

Their distribution network consists of farmers F, collection sites C, production facilities P, and supermarkets S, with $V = C \cup P \cup S$. The coordinates of the different locations can be found in file `coordinates.xlsx`, and supply of raw milk, production capacities, and demand per dairy product can be found in file `supply_cap_demand_data.xlsx` (unit: 100 liters for raw milk and 100kg for dairy products). Each truck can carry 800 liters or 800 kg. At production facility $P1$, 100l raw milk are used for producing 50kg of milk, 20kg of yogurt, and 30kg of cream. At production facility $P2$, 100l of raw milk result in 60kg of milk, 10kg of yogurt, and 30kg of cream.

1 Transporting Dairy Products from Collection Sites to Supermarkets

Help FreshMilk to set up its distribution network, also considering how the network could develop in the future. Throughout this exercise, we ignore the collection of milk from the farmers and only focus on the transport from the collection sites to the production facilities to the supermarkets.

1.1. Preparation Steps (7 points)

From the data FreshMilk provided you, you cannot directly obtain all the data you need.

(a) (2 points) Visualize the network of FreshMilk.

(b) (3 points) Calculate variable costs. The costs depend both on a salary for the driver and consumption of gas for the truck. The drivers receive 20€/hour, and the velocity of a truck is 25km/h within cities and 60km/h outside cities. Gas currently costs 0.5€/km. Haversine distances have proven to be a good heuristic for road distances.

Provide a `varCosts.xlsx` table for variable costs. Variable costs are proportional to transporting 100 liters of raw milk or 100kg of dairy product, i.e., one unit in the files. Provide a file `ex1_distanceCalc.py` which contains a function for calculating variable costs for each pair of locations.

(c) (2 points) Calculate fixed costs. Each direct connection comes at a fixed cost of 100€ for using the truck, and 1.0€/km (using Haversine distances) for deprecation. Provide a `fixedCosts.xlsx` table for fixed costs. Provide a file `ex1_distanceCalc.py` which contains a function for calculating fixed costs for each pair of locations.

1.2. Basic Model (18 points)

The previous intern of FreshMilk has developed a mathematical model that allows the company to decide **how many** dairy products to send from which collection site $i \in C$ to which production facility $j \in P$ to which supermarket $k \in S$, minimizing the variable cost (henceforth referred to as *Model 0*):

$$\min \sum_{i \in C} \sum_{j \in P} c_{ij} x_{ij} + \sum_{i \in P} \sum_{j \in S} c_{ij} x_{ij} \tag{1}$$

$$\sum_{j \in P} x_{ij} \leq s_i \qquad \forall i \in C \tag{2}$$

$$\sum_{j \in C} x_{ji} = \sum_{j \in S} x_{ij} \qquad \forall i \in P \tag{3}$$

$$\sum_{j \in P} x_{ji} = d_i \qquad \forall i \in S \tag{4}$$

$$x_{ij} \geq 0 \qquad \forall i, j \in V \tag{5}$$

(a) (3 points) Implement the above model in Python and Gurobi (name this file `ex1_model0.py`).

(b) (3 points) Is this solution feasible for FreshMilk's actual problem? Is this solution optimal? Explain your answer. Provide insights into when the solution would be feasible/optimal and when not.

(c) (5 points) The above model does not account for the possibility of transshipment between collection sites and between production facilities. Extend the mathematical model (now called Model 1) and your implementation to include this aspect (name this file `ex1_model1.py`).

(d) (2 points) How much can FreshMilk gain from the additional flexibility due to transshipment? Does the transshipment between collection sites or the transshipment between production facilities help more?

(e) (5 points) FreshMilk currently does not consider different dairy products and associated capacities. Adapt Model 1 and the implementation to account for raw milk transport from collection sites to production facilities and dairy product transport from production facilities to supermarkets. This new model is Model 2 (name the Python code `ex1_model2.py`).

1.3. Improving Dairy Distribution (15 points)

The actual problem of FreshMilk is more intricate than Model 2 since trucks incur fixed costs. Further, FreshMilk is interested in understanding the key drivers of their costs.

(a) (5 points) Extend Model 2 and its implementation by determining the number of trucks on each arc, considering fixed costs per truck. Name this new model Model 3 (and the file `ex1_model3.py`).

(b) (2 points) Compare the transportation plan and actual costs with and without fixed costs (i.e., Model 2 and Model 3). How does the transportation plan change in the presence of fixed costs? Explain the difference and visualize both transport plans.

(c) (4 points) How do the transportation plan and the total costs change if the demand for yogurt increases while the demand for the other dairy products remains the same? Create a function in `ex1_model3.py` to test this, provide a graph visualizing the costs, and explain your findings.

(d) (4 points) How do the optimal fixed and variable costs change if the production capacity of production facility $P2$ changes? Create a function in `ex1_model3.py` to test this, provide a graph visualizing the costs, and explain your findings.

1.4. Adapting Distribution to Fluctuations in Supply (10 points)

The cows at the farms provide different amounts of raw milk each day, resulting in different supply at the collection sites, s_i^ω where $\omega \in \Omega$ is a scenario (a realization of the actual supply). You can find 10 scenarios in file `supply_scenarios.xlsx`. FreshMilk can adapt how they fill the trucks, but the number of trucks must not change on a day-to-day level. FreshMilk is interested in how supply randomness influences their decisions.

(a) (3 points) How does your network design perform if supply varies? Use a function in `ex1_model3.py` to automate the check.

(b) (7 points) Develop and implement a mathematical model called Model 4 which determines the optimal network design considering all supply scenarios Ω. How does this network design differ from the network design to Model 3? Name the Python code file `ex1_model4.py`.

2 Collecting Milk from Farms

Every morning, the milk trucks of FreshMilk collect dairy from farms nearby. In the files `time_between_farms_Cx.xlsx`, you can find the travel time between the different farms served from collection sites $C1$ and $C2$ (in hours). The file `pickuptime_per_farm_Cx.xlsx` lists the preferred time FreshMilk collects milk from each farm (in minutes since the start of the time horizon.

2.1. Basic Modelling and Implementation (20 points)

You can use two different formulations for the TSP, the *Miller-Tucker-Zemlin* formulation (MTZ) and the *Dantzig-Fulkerson-Johnson* formulation (DFJ). Both have their advantages and disadvantages.

(a) (4 points) Provide both mathematical models for MTZ and DFJ.

(b) (6 points) Implement the TSP with the MTZ formulation in Python and Gurobi. Name the file `tsp_MTZ.py`.

(c) (6 points) Implement the TSP with the DFJ formulation in Python and Gurobi. Name the file `tsp_DFJ.py`.

(d) (4 points) Which of the formulations is faster? Why? Does this differ between the different demand patterns?

2.2. **Improving DFJ** (15 points)

(a) (5 points) Implement a function that, for a given solution, determines if it contains subtours. Name the file `tsp_DFJ_BC.py`.

(b) (5 points) Extend your implementation of the DFJ formulation such that it only eliminates those subtours that actually emerge for a given instance. Add to the file `tsp_DFJ_BC.py`.
(Hint: Let Gurobi solve the instance entirely and then iteratively add those constraints that are violated.)
(Hint: If you were not able to solve question 2.2(a), you can manually separate violated constraints and still earn points here. Use comments for each iteration, to show how to sequentially add the subtours.)

(c) (5 points) How does the runtime of the new implementation of DFJ compare with the runtime of your old implementation of DFJ, and the implementation of MTZ? Why? Does this differ between demand patterns?

2.3. **Timing of Collection** (15 points)

So far, you consider a fairly simple TSP. Of course, collecting raw milk is a more complex endeavour. Due to the operations of the farms, they cannot provide milk at all times, but only after they finished milking the cows. Due to storage restrictions, the milk must then be collected as quickly as possible, preferably within 30 minutes. Later collection comes at a risk of the milk turning bad, approximated by a cost of €10 per hectoliter and hour (relative to routing costs of €0.5 per minute).

(a) (3 points) Extend the MTZ formulation to consider (soft) time windows for delivery.

(b) (4 points) Extend your implementation of the MTZ formulation to consider (soft) time windows. Name the file `tsp_MTZ_TW.py`.

(c) (3 points) How does the width of the time window impact the collection efficiency (route duration and total costs)?

(d) (5 points) To not spoil any raw milk, the trucks must drop off milk at a collection point no later than 2 hours after collection. However, the same truck can do multiple tours, and can also travel between the two collection sites. Develop a mathematical model that includes this additional flexibility.

3 Workload reporting

3.1. **Workload reporting** (1 point)

Please truthfully report

(a) who did which work

(b) how much work was it in total (in detail)

(this is a bonus point)

Reporting

Hand in the assignment via Canvas. Upload a **.pdf** file containing the complete report with answers to all questions (**NO MS Word documents**). Furthermore, hand in a .zip file containing your full Python code as well as Excel files you hand in. Make sure that your report (PDF) is both inside the zip archive and uploaded separately! Include assignment number, group number, and your last names in the filenames, e.g., `Assignment1_Group3_Martin_vanLieshout_Chen.pdf` and `Assignment1_Group03_Martin_vanLieshout_Chen.zip`. All your Python scripts must be executable irrespective of your computer, and must produce clear output. The front page of your report must state your group number, the names and student IDs of every group member. The final report cannot exceed 8 pages (including pictures etc., but excluding the front page). Deviation of these requirements will reduce your final grade by 1 point (out of 10).