FreshMilk

Assignment 1

30-11-2023

Tomas Henriquez – 2047934 & Sam Veld - 1466178
Eindhoven University of Technology
1CM110 – Decision-making in Transport and Mobility

EINDHOVEN UNIVERSITY OF TECHNOLOGY

Table of Contents

1. Transporting Dairy Products from Collection Sites to Supermarkets

1.1 Preparation Steps

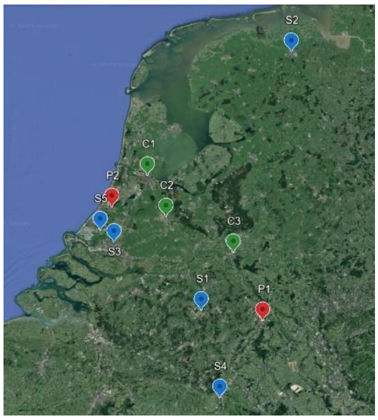

Figure 1: Visualization of the FreshMilk network (Google Earth, 2023)

As an initial step, the network of FreshMilk is visualized as can be seen in figure 1. The coordinates established in the coordinates.xlsx file were put in the Google Earth visualization tool to provide the following visualization. Collection sites C1, C2, and C3 are indicated with green and located respectively in Amsterdam, Utrecht, and Nijmegen. Production sites P1 and P2 are located respectively in Venlo and Leiden. Supermarket sites S1, S2, S3, S4, and S5 are located respectively in Eindhoven, Groningen, Rotterdam, Maastricht, and Delft. All facilities are located in the Netherlands and quite well accessible.

The next preparation step has to do with the calculation of the variable costs. These costs depend on the salary of the driver and the gas consumption of the truck. As explained in the assignment, the drivers receive 20 €/hour and the velocity of a truck is 25 km/h within cities and 60 km/h outside cities. Additionally, gas costs 0.5 €/km. All these values are incorporated in the "ex1_distanceCalc.py" file, from which the code can be found in Appendix 1.1. The file contains a function for the calculation of the variable costs for each pair of locations. The output file contains a table for variable costs in the "varCosts.xlsx" file that can be seen in Appendix 1.1B.

There is an important assumption that should be considered within this question. The time and distance driven inside a city should be assumed a certain value. The considered value for this assignment would be 5km to enter the city and 5 km to exit the city. These assumptions are based on research and by analyzing the locations on the map. Within the city, trucks have to drive slower for a certain distance and time.

The following preparation step requires the calculation of fixed costs. Fixed costs consist of €100 for using a truck on a direct connection and 1.0 €/km for deprecation with the use of Haversine distances. These fixed cost values are integrated in the "ex1_distanceCalc.py" file, from which the code can be found in Appendix 1.1. The file contains a function for the calculation of the fixed costs for each pair of locations. The output file contains a table for fixed costs in the "fixedCosts.xlsx" file that can be seen in Appendix 1.1C.

1.2 Basic Model

To minimize the variable cost, FreshMilk can decide how many dairy products to send from which collection site i ∈ C to which production facility j ∈ P to which supermarket k ∈ S, according to a mathematical model that can be seen in figure 2.

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

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

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

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

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

Figure 2: Mathematical model 0

The mathematical model is a min-cost-flow problem. The first line in the model contains the objective function, which strives for a minimization of the cost on the sum of all flows from collection sites C to production facilities P and from production facilities P to supermarkets S. The second line contains a constraint that shows that the flow should be lower or equal to the supply. The third function includes a constraint that makes sure that the input flow coming from the collection sites should be equal to the output flow arriving at supermarkets. The fourth line indicates that the flow should be equal to the demand. Finally, the fifth constraint indicates that the flow could not be negative and that it should be equal or larger than zero.

1.2A

The following step is to convert this mathematical model into a python code, in order to be able to do calculations with it. The implementation of the model in Python and Gurobi is stated in the "ex1_model0.py" file that can be found in Appendix 1.2A.

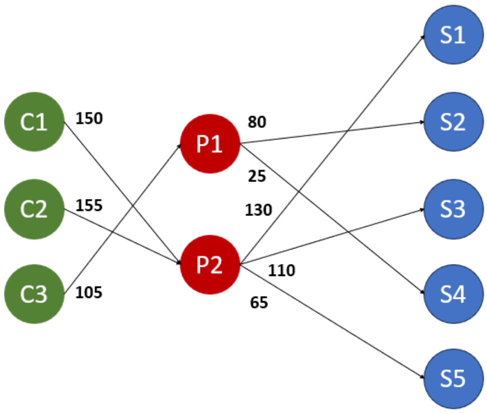

Figure 3: Network model 0

With the limited constraints in this model, a solution is identified which moves the majority of product through P2. No collection site supplies both production facilities and no supermarket receives product from both production facilities.

1.2B

The solution that results from mathematical model 0 is not feasible, since the capacity for P2 is exceeded (305 collected when P2 has a capacity of 250). Additionally, it is not optimal as very many factors are not considered, such as the possibility for transshipments, fixed costs and number of trucks needed to ship products, and distinctions across products.

1.2C

In this step, mathematical model 0 is extended to a new version, namely mathematical model 1. In this new model, the transshipments are taken into account. Transshipment, in this situation, is the possibility of shipments between locations of the same type, for instance, a shipment between two collection sites or between two production facilities. Constraints for the capacities for the production sites were also considered in this model.

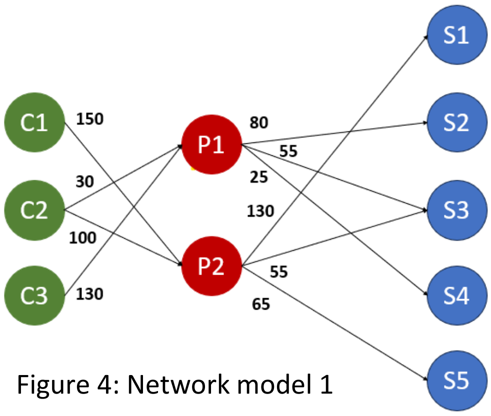

Figure 4: Network model 1

In this case, distribution differs because the production facilities must adhere to their capacities, which is best illustrated by the change in P2 production from 305 in the previous model to 250 in this model.

1.2D

This model showed that transshipments across collection sites and across production facilities are not favorable and do not serve to reduce the overall cost of the solution in this scenario. Shortly, FreshMilk can not gain much from the additional flexibility due to transshipments.

1.2E

Mathematical model 1 could be improved even further by considering different dairy products and their associated capacities. Still, raw milk will be transported from collection sites to production facilities. However, in this case, within the production facilities, the raw milk will be used to produce different dairy products that are shipped from the production facilities to the supermarkets.

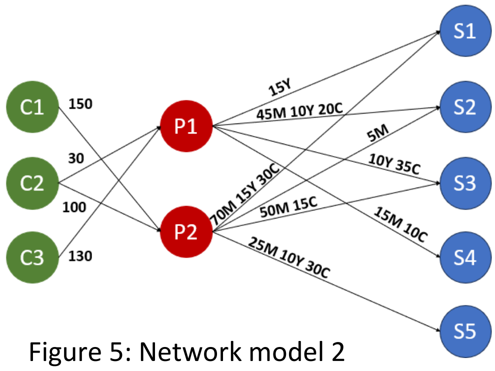

Figure 5: Network model 2

In this model, production facilities must adhere to their capacities for each product, which creates some interesting delivery situations to the supermarkets. For instance, S2 is supplied almost entirely by P1 but 5 units of milk must be delivered from P2 to optimize while adhering to the capacity for milk production by both P1 and P2. This is inefficient as a half empty truck must be deployed from P2 to S2, but this model does not account for the costs associated with shipment, only travel distance.

1.3 Improving Dairy Distribution

1.3A

Starting from this step, the mathematical model is improved to be more realistic and related to the actual problem of FreshMilk. Mathematical model 3 is coded in the “ex1_model3.py” file and can be found in Appendix 1.3A. It includes an assumption for the number of trucks on each arc, while considering fixed costs per truck.

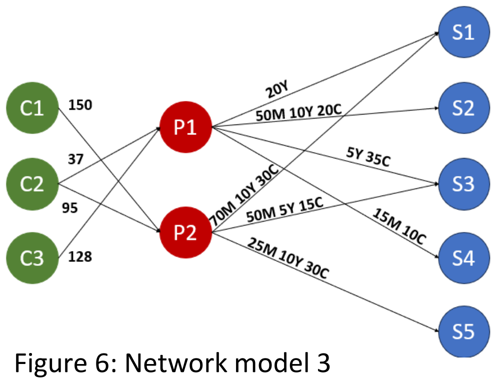

Figure 6: Network model 3

1.3B

Mathematical model 2 lacks the fixed costs, while mathematical model 3 includes them. This makes a huge difference in analyzing the problem. The transportation plan changes when the fixed costs are present as the model must now optimize for full or mostly full trucks to minimize the cost of transit per unit of product. As such, the half full trucks that would be present in model 2’s solution are largely avoided by model 3, opting instead to get as close to multiples of the truck capacity (8 units) while adhering to the supply, capacity, and demand restrictions for each location. The diagram of the solution by model 2 can be seen in section 1.2E while the diagram of the solution by model 3 can be seen in section 1.3A.

1.3C

The transportation plan and the total costs are changing if the demand for yogurt increases while the demand for other dairy products remain the same, as can be seen in the “model3_sensitivity_results.xlsx” file.

This can be tested with a new function in the “ex1_model3.py” file. As can be seen in figure 7, an increase in yogurt demand increases the costs until the solution is not feasible anymore. After an increase factor of 1.5, there is no feasible solution anymore.

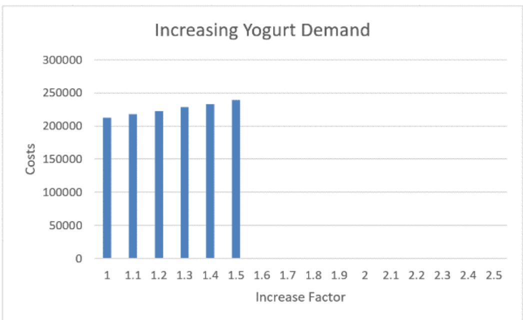

Figure 7: Increasing yogurt demand

1.3D

When the production capacity of production facility P2 in Leiden changes, there are major changes in the optimal fixed and variable costs, as can be seen in the "model3_sensitivity_results.xlsx" file.

This can be tested with another new function in the "ex1_model3.py" file. As can be seen in figure 8, an increase in the production capacity of P2, decreases the total costs.

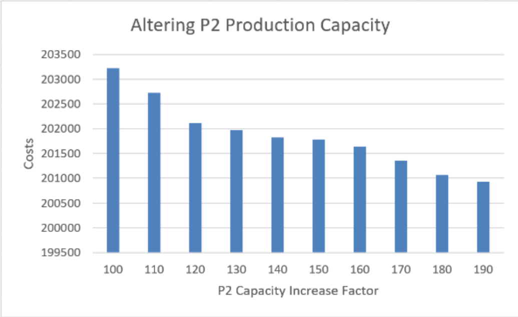

Figure 8: Altering P2 Production Capacity

1.4 Adapting Distribution to Fluctuations in Supply

1.4A

Another new function in the "ex1_model3.py" file, can be used to test how the network design performs when the supply varies. The results can be seen in the "model3_sensitivity_results.xlsx" file. As can be seen in figure 9, each supply scenario causes a different cost. What stands out is that supply scenario 10 results in the highest cost, while supply scenario 6 results in the lowest cost. The maximum cost difference within the supply scenarios is approximately €6000.

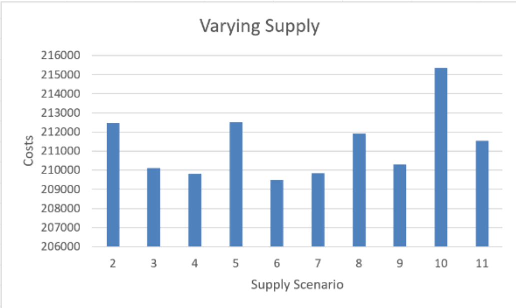

Figure 9: Varying Supply

1.4B

To build a model which takes into account all supply scenarios, we modified our existing constraint for the supply (constraint 1 in figure 2) as follows:

$$\sum_{j \in P} x_{ij} \leq s_i^{\omega} \qquad \forall i \in C, \forall \omega \in \Omega$$

Where Ω is the set of all supply scenarios.

Model 4 was incapable of identifying any single feasible solution, as it is forced to consider the worst possible supply (lowest supply) for all scenarios for each collection site. This combination of minimums does not produce enough supply to meet the demands of the supermarkets. This implies that FreshMilk should optimize their collections based on the supply of the day, as there is no single solution which optimizes supply pickup across all scenarios.

2. Collecting Milk from Farms

2.1 Basic Modelling and Implementation

In general, the milk trucks of FreshMilk collect dairy from farms nearby. This means that there are travel times between the different farms served from collection sites C1 and C2. We must address the preferred time that FreshMilk should be collecting milk from each farm. In general, this complex situation forms a type of Travelling Sales Person (TSP) problem. The TSP could consist of two separate formulations, the Miller-Tucker-Zemlin (MTZ) and the Dantzig-Fulkerson-Johnson (DFJ) formulations.

2.1A

Firstly, it is useful to have a look at the mathematical model for the MTZ as can be seen in figure 10.

$$
\begin{array}{l}
\text{Minimize:} \quad \sum_{i \neq j} c_{ij} x_{ij} \\
\\
\sum_{i \neq j} x_{ij} = 1, \quad \forall j \quad (1) \\
\sum_{i \neq j} x_{ji} = 1, \quad \forall i \quad (2) \\
\\
u_i - u_j + n x_{ij} \leq n - 1, \quad \forall i \neq 1, j \neq 1, i \neq j \quad (3) \\
\\
x_{ij} \geq 0, \quad \forall i, j \quad (4) \\
u_i \geq 0, \quad \forall i \quad (5)
\end{array}
$$

Figure 10: Mathematical model – MTZ

The first line is indicating the objective function, which is aiming for the minimization of the total cost of the tour. Then, the first constraint is making sure that each location is visited exactly once and the second constraint is making sure that each location is left exactly once. Additionally, the third constraint contains the MTZ constraints that eliminate sub-tours by assigning position variables u_i to each location. These constraints ensure that if $x_{ij} = 1$, then $u_i - u_j + nx_{ij}$ is less than n-1. Finally, the fourth and fifth constraint are non-negativity constraints which make sure that the travel indicator x and tour order u are equal or greater than zero. x_{ij} is a binary variable indicating whether there is directly travelled between location i and j. A value of 1 indicates a direct travel from location i to j and 0 otherwise. u_i is a continuous variable representing the order or position of location i in the tour.

Secondly, it is relevant to analyze the mathematical model for the DFJ as can be seen in figure 11.

$$
\begin{array}{l}
\text{Minimize:} \quad \sum_{i \neq j} c_{ij} x_{ij} \\
\\
\sum_{i \neq j} x_{ij} = 1, \quad \forall j \quad (1) \\
\sum_{i \neq j} x_{ji} = 1, \quad \forall i \quad (2) \\
\\
u_i - u_j + n x_{ij} \leq n - 1, \quad \forall i \neq 1, j \neq 1, i \neq j \quad (3) \\
\\
z_{ij} + z_{ji} \geq 1, \quad \forall i \neq 1, j \neq 1, i \neq j \quad (4) \\
z_{ij} \in \{0, 1\}, \quad \forall i \neq 1, j \neq 1, i \neq j \quad (5) \\
\\
x_{ij} \geq 0, \quad \forall i, j \quad (6) \\
u_i \geq 0, \quad \forall i \quad (7)
\end{array}
$$

Figure 11: Mathematical model – DFJ formulation

The first line is stating the objective function, which is aiming for the minimization of the total cost of the tour. Then, the first constraint is making sure that each location is visited exactly once and the second constraint is making sure that each location is left exactly once. In addition, the third, fourth, and fifth constraint are the DFJ sub-tour elimination constraints that prevent the formation of subtours. In the end, the sixth and seventh constraints form the non-negativity constraints. Again, x_{ij} is a binary variable indicating whether there is directly travelled between location i and j and u_i is a continuous variable representing the order or position of location i in the tour. Additionally, z_{ij} is a binary variable indicating the subtour elimination variable. A value of 1 indicates if a sub-tour between locations i and j is eliminated and 0 otherwise.

Overall, the MTZ and DFJ formulations are quite similar, but the main difference lies in the fourth and fifth constraint of the DFJ formulation. While both formulations prevent sub-tours, the key difference lies in how they express the sub-tour elimination conditions. MTZ uses the position variables u_i directly in the constraints, whereas DFJ introduced binary subtour elimination variables z_{ij} to achieve the same goal.

2.1B

As a beginning step, the TSP with the MTZ formulation is implemented in Python and Gurobi with the "tsp_MTZ.py" file. The output contains a tour from collection site C1 to each farm and a tour from collection site C2 to each farm, as can be seen in the figures below.

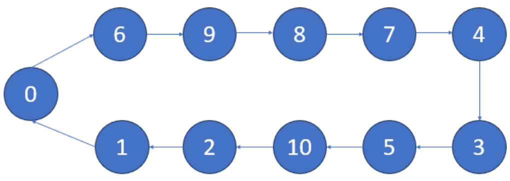

Figure 12: MTZ C1

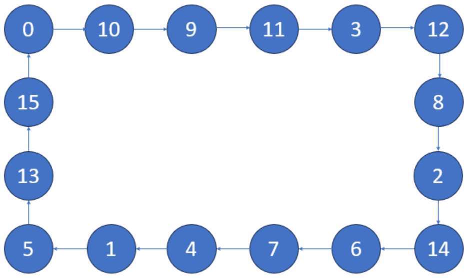

Figure 13: MTZ C2

2.1C

Subsequently, the TSP with the DFJ formulation is implemented in Python and Gurobi with the "tsp_DFJ.py" file. The only difference from the MTZ formulation takes place in C2, as can be seen in the figures below.

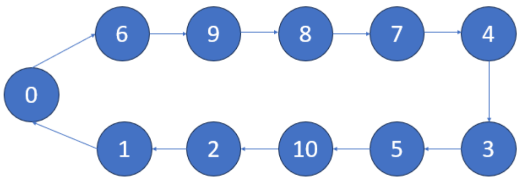

Figure 14: DFJ C1

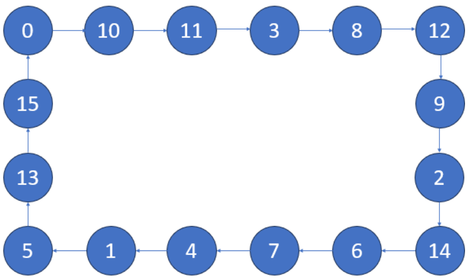

Figure 15: DFJ C2

2.1D

The MTZ formulation appears to perform faster than the DFJ formulation. This makes sense as the two formulations are largely the same, but MTZ has fewer constraints and decision variables to consider.

2.2 Improving DFJ

2.2A

A function that determines if a given solution contains subtours can be found in "tsp_DFJ_BC.py" file.

2.2B

Unfortunately, due to time, we were not able to complete this portion of the assignment.

2.2C

The runtime of the new DFJ implementation can vary compared to the old DFJ and MTZ formulation due to differences in constraints and solution approaches. The addition of subtour elimination constraints in DFJ may enhance accuracy but potentially lead to longer runtimes, specifically in scenarios with complex demand patterns. MTZ might exhibit faster convergence. Additionally, the impact of demand patterns on runtime is significant, with DFJ potentially being more sensitive to complex subtour structures.

2.3 Timing of Collection

2.3A

The original MTZ formulation is adapted with more constraints and a new objective function, as can be seen in the figure. In these constraints, t_i constitutes the arrival time at location i, while t_ℓ symbolizes the final arrival time, c_R the routing costs, ℓ_i the amount of late time for location i, c_P the penalty cost for a late arrival, and $w_{\ell i}$ and w_{ui} represent the lower bound and upper bound for the pickup time at location i, respectively.

$$\text{Minimize: } t_\ell \times c_R + \sum_i \ell_i \times c_P$$
$$1)\ \sum_{i \neq j} x_{ij} = 1, \ \forall j$$
$$2)\ \sum_{i \neq j} x_{ji} = 1, \ \forall i$$
$$3)\ u_i - u_j + n x_{ij} \leq n-1, \ \forall i \neq 1, j \neq 1, i \neq j$$
$$4)\ x_{ij} \geq 0, \ \forall i,j$$
$$5)\ u_i \geq 0, \ \forall i$$
$$6)\ t_j \geq (t_i + c_{ij}) \times x_{ij}, \ \forall i \neq 1, j \neq 1, i \neq j$$
$$7)\ t_j \geq c_{ij} \times x_{ij}, \ \forall j, i = 1$$
$$8)\ t_\ell \geq (t_i + c_{ij}) \times x_{ij}, \ \forall i, j = 1$$
$$9)\ \ell_i \geq t_i - w_{ui}, \ \forall i \neq 1$$
$$10)\ \ell_i \geq 0, \ \forall i \neq 1$$
$$11)\ t_i \geq w_{\ell i}, \ \forall i \neq 1$$

Figure 16: Extended MTZ

The additional constraints come in the form of 6) ensuring that the arrival time at j is greater than or equal to the arrival at i plus the travel time from i to j for chosen links. Special conditions 7) and 8) ensure that this is accurate for leaving from and arriving to the collection site, where arriving to the collection site signifies the final arrival time. 9) calculates the lateness factor as it indicates the difference between actual arrival and the upper bound for the pickup time at location i, while 10) ensures that lateness cannot be negative. Finally, 11) ensures that pickup at location i cannot take place before its pickup lower bound time.

2.3B

The optimal Hamiltonian cycle produced by this formulation differs significantly from the basic MTZ formulation. Here, the problem focuses on minimizing costs by optimizing collection times and avoiding the penalty associated with collection outside of the specified time window. Here the cost of traveling a longer distance is significantly less important (EUR0.5 vs EUR10).

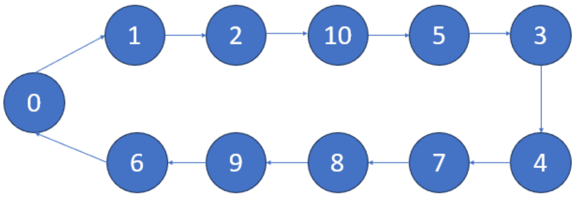

Figure 17: C1

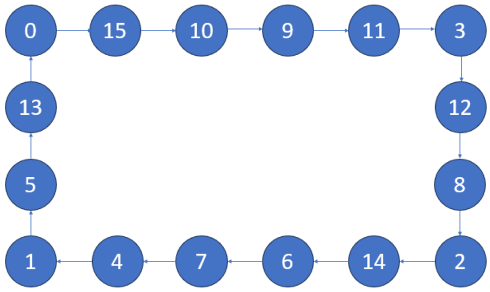

Figure 18: C2

2.3C

Shortening the time window leads overall to higher costs, while lengthening the window leads to lower overall costs. This is because with shorter windows there are less opportunities to avoid incurring late pickup penalties, while the opposite is true for longer windows of time. Travel times, on the other hand, are nearly equivalent. However, decreasing time windows makes the model less likely to travel further to adhere to pick-up times and more likely to focus on traveling shorter distances to minimize the overall lateness across all collections. Therefore, we might see slightly lower travel times for smaller time windows.

2.3D

Unfortunately, due to time, we were not able to complete this portion of the assignment.

3. Workload reporting

3.1 Workload reporting

In this assignment spanning 2.5 weeks, both team members focused on developing the logic behind the solutions together. However, Tomas took the lead in most programming tasks, dedicating significant effort to improving the project's technical aspects. Simultaneously, Sam played a crucial role in ensuring the assignment's clarity and maintaining the organization in terms of document structure and question presentation. The collaboration was of high quality and it increased especially after the third team member left the group. Close to the end of the process, Sam also invested some time in refining the report's structure and layout, while Tomas delved into sourcing background knowledge related to specific challenges in the code. Despite the challenges, the teamwork remained strong, showcasing effective collaboration even amid unexpected disruptions.

The assignment demanded substantial time and effort, with both Tomas and Sam consistently contributing approximately 15 hours per week. This collective dedication resulted in nearly 75 hours invested over the entire duration of the project. The challenges were heightened by the need for an understandable review of background knowledge before starting the assignment, emphasizing the depth of the undertaken tasks. In addition, the difficulties escalated when one of the team members was not performing at all, which added an extra layer of complexity and necessitated additional efforts to navigate and overcome the resultant struggles.