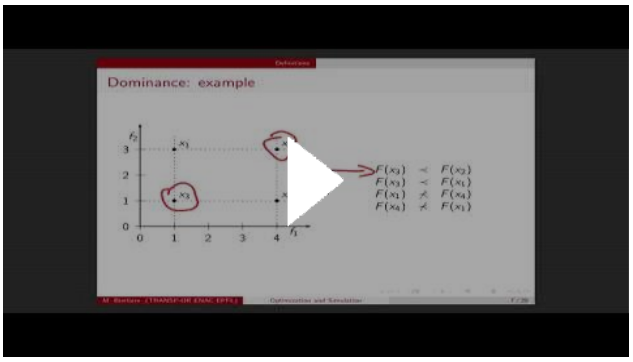
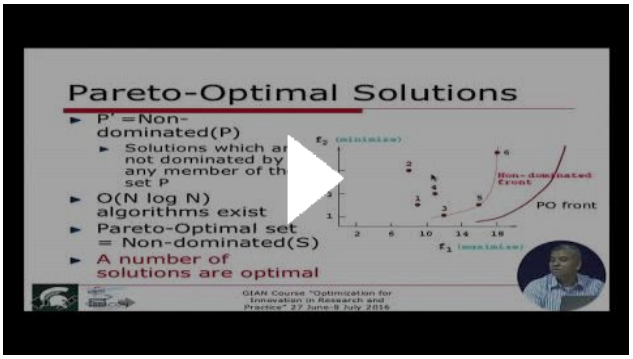
First Assignment

Samstag, 22. Mai 2021 09:39

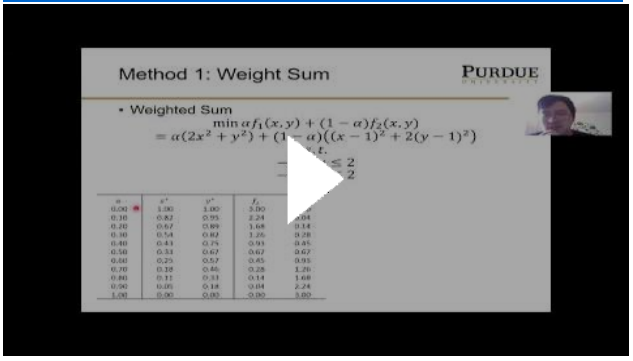
1. **Assignment 1 in Latex**
   https://www.overleaf.com/project/609a5574e7374883e5d3f3b5

2. https://developers.google.com/optimization/routing/vrp
3. https://www.sciencedirect.com/topics/computer-science/weighted-sum-method#:~:text=One%20of%20the%20most%20commonly,generate%20multiple%20Pareto%20optimal%20solutions%2C
4. Optimization and simulation. Multi-objective optimization - part 1
5. 

6. https://de.mathworks.com/help/optim/ug/generate-and-plot-a-pareto-front.html
7. **Soft constraint**: another approach for solving supplying problem with preference extension recommended from Xie!!! --> try it later
   https://or.stackexchange.com/questions/1050/soft-constraints-and-hard-constraints
8. **Pareto optimal set**
   23. Multiobjective Optimization

9. Good for looking for Pareto-optimal set
10. MET 503 Lecture 18: Multi-Objective Optimization Problem

11. Soft constraint
12. https://www.gams.com/latest/docs/UG_EMP_SoftConstraints.html
13. Non-dominated sorting genetic algorithm 21:08
    https://www.youtube.com/watch?v=Hm2LK4vJzRw
14.

In order to ensure that there is only one truck travelling on one arc at a time, one more decision variable $s_{ij}^{vv'k}$ is defined. $s_{ij}^{vv'k} = 1$ if truck v travelling on arc (i, j) before truck v', and it takes value 0 otherwise. Using big M technique, this constraint is presented as follows:

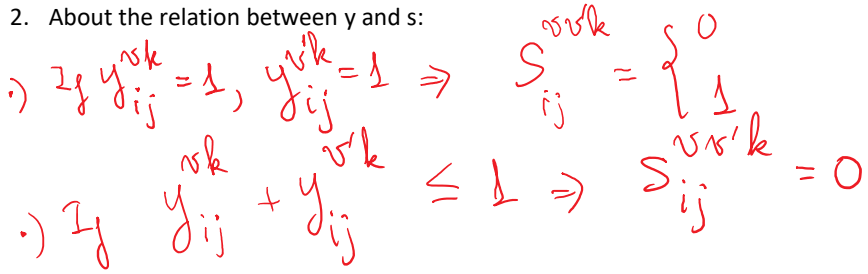
- **Only one truck travelling on one arc at a time**

$$w_j^{vk} \le w_i^{v'k} + M(2 - y_{ij}^{vk} - y_{ij}^{v'k}) + M(1 - s_{ij}^{vv'k}), \quad (i,j) \in A, v, v' \in V, k \in K \quad (1.16)$$

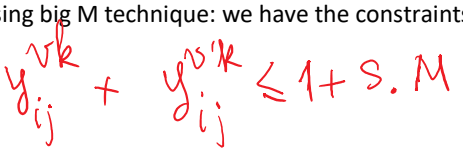
1. Because truck v going on arc (i,j) before truck v', the time windows for two trucks have to be satisfied the following time axis

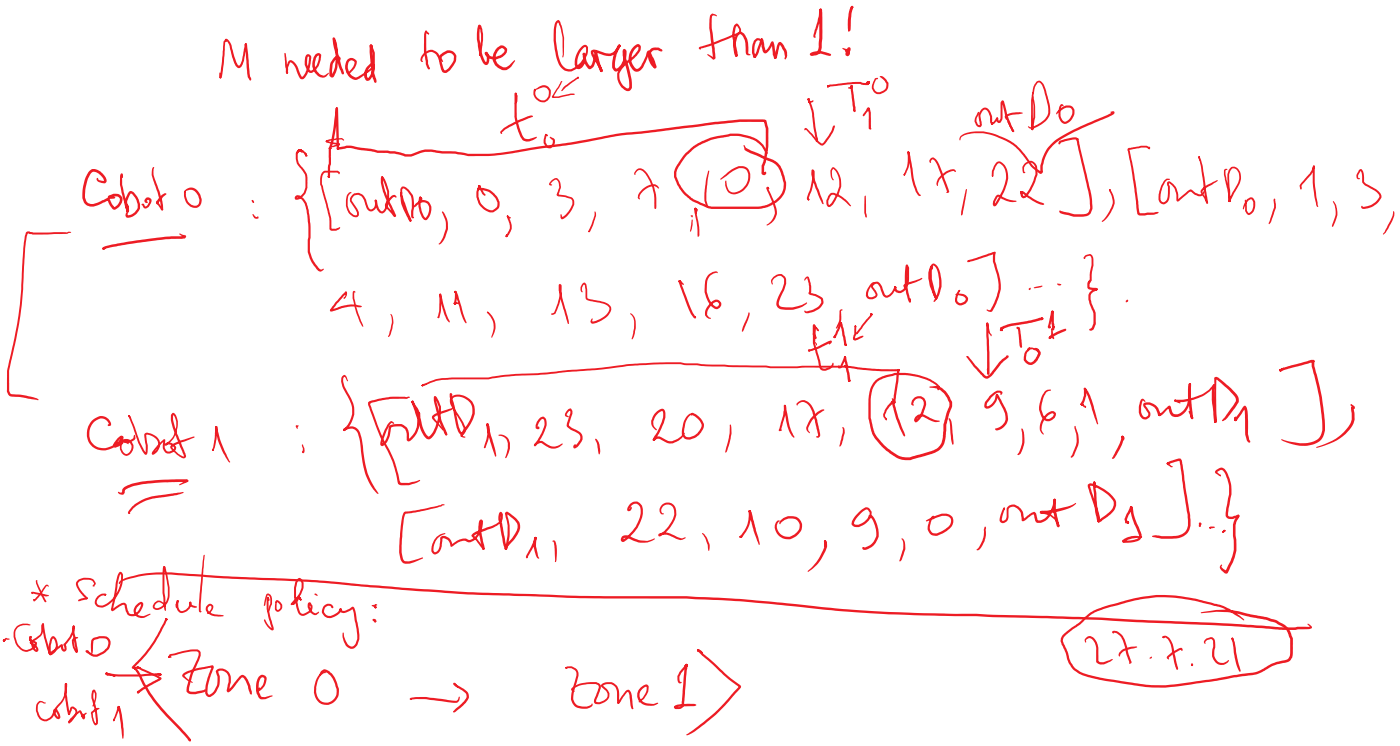
To ensure that the 2 time intervals (windows) should not be overlapped.

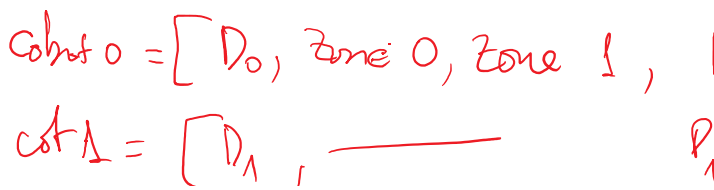
2. About the relation between y and s:

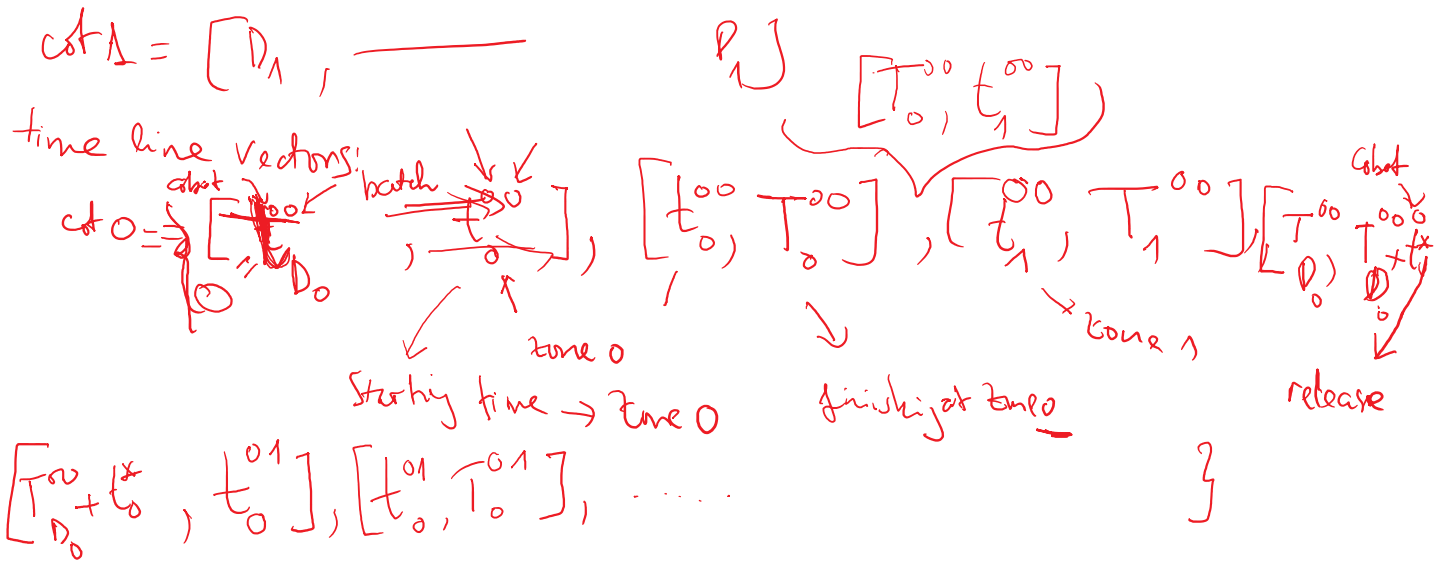


Using big M technique: we have the constraints for y and s is:



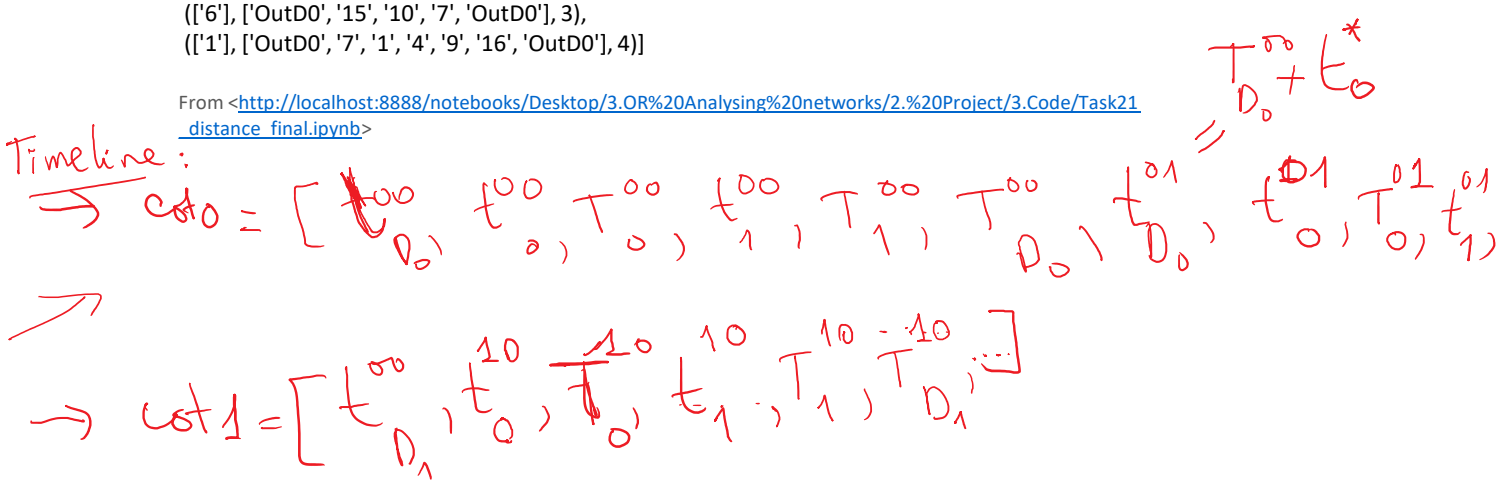




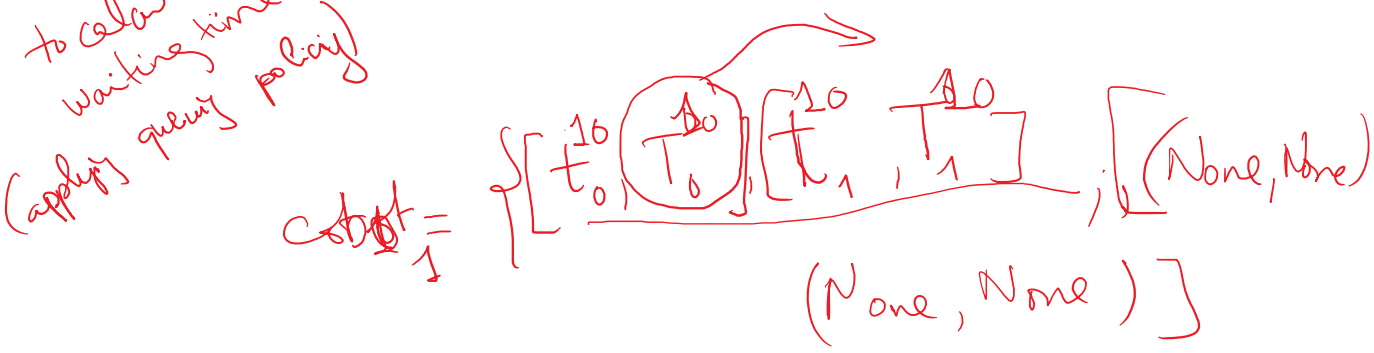


[(['2', '0', '3'], ['OutD0', '1', '2', '10', '9', '15', 'OutD0'], 1),
(['8'], ['OutD0', '15', '16', '11', '2', '7', '1', 'OutD0'], 2),
(['6'], ['OutD0', '15', '10', '7', 'OutD0'], 3),
(['1'], ['OutD0', '7', '1', '4', '9', '16', 'OutD0'], 4)]

From <http://localhost:8888/notebooks/Desktop/3.OR%20Analysing%20networks/2.%20Project/3.Code/Task21_distance_final.ipynb>



Cobot1 and cobot2
Could be lists of lists

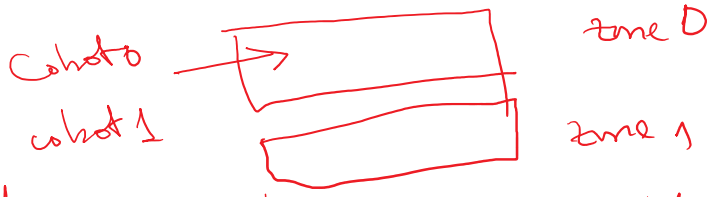


How to calculate the waiting time and the time points in the time line?

How to calculate the waiting time and the time points in the time line?

1. Calculate those $t, T, T_{D_0}, T_{D_1}, t_{D_0}, t_{D_1}$ without waiting time
→ need to extract shelves in zone 0, 1 for each batch.
2. Query policy:
→ zone 0 → zone 1

Keep the order of the shelves to Calculate the initial timeline Without waiting time



3. if else statement for the first pair of intervals of cobot 0 and cobot 1:

if $(t_0^{00} \leq t_0^{10})$:

$\text{cobot}_1[1:] := T_0^{10} \mathrel{+}= T_0^{00} - t_0^{00}$ (serving time in zone 0 for cobot 0)

$\text{cobot}_1[2:] \mathrel{+}= T_0^{00} - t_0^{00}$ (from the time point T_0^{10} on, timeline of cobot 1 is updated by the serving time in zone 0 for cobot 0)

else:

$\text{cobot}_0[1:] \mathrel{+}= T_0^{10} - t_0^{00}$

4. Working with the loops and the two timeline vectors of the cobots

4. a) make the two cobot vectors same length

len

Reminder

If we have a tour like this

([1'], ['OutD0', '7', '1', '4', '9', 'OutD0'], 4)]

zone 0

→ Then the interval for zone 1 of this tour would be (None, None)

Reminder

a2: (['2', '0', '3'], ['OutD0', '1', '2', '10', '9', '15', 'OutD0'], 1),

← subsection 4.1

(< (which items → subsection 4.2)
How many items picked from this shelf)

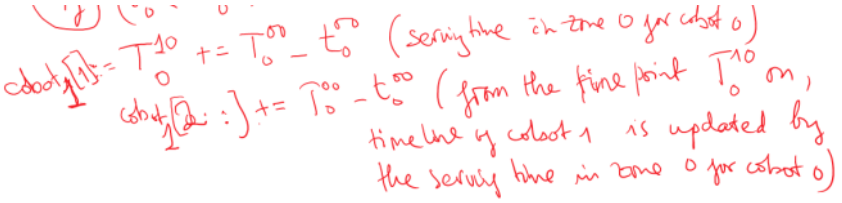
→ Remember to save the quantities of the items in the step of eleminating dublication in the Function greedyTour

b) Loop over the batches (or tours)

For batch_0 in cobot0:

3. if else statement for (if) $(t_0^{00} \leq t_0^{10})$:

for batch_1 in cobot1:
if batch_0[0][0] <= batch_1[0][0] (see 3. in if statement):
(update)



5. Structure of the project

- Task 2.1: generate the initial solution s. Please accomplish the last function with for loop for the depots.
- Task 2.2: generate a neighborhood of s (subsection 4.2) + local search: compare f(s) and f(s'), where s' is from the neighborhood of s (subsection 4.3 in paper, set some maximal number of iterations): we need to calculate the makespan(f).
- Task 2.3: would be difficult: mixed shelf policy.