INC 364 - 2021

Gurobi Optimization – Part 1

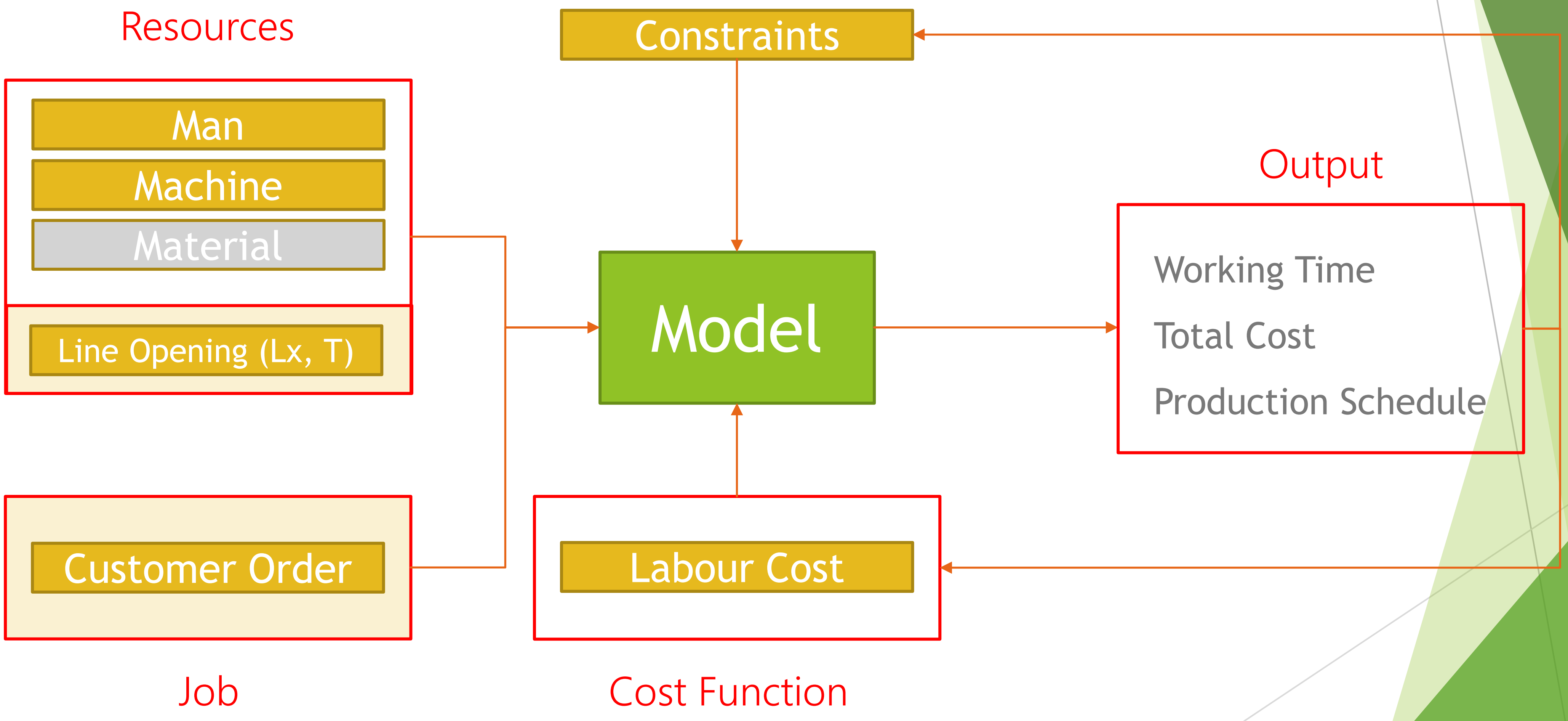
What is Mathematical Optimization?

- Mathematical Programming, can help to answer the question, “What should we do?”.

- It turns a business problem into a math model and then finds the best solution.

- The Gurobi Optimizer captures the key features of your business problem in a mathematical optimization model, and automatically generates an optimal solution.

Part 1.1 - Workshop Problem



Part 1.1 - Workshop Problem

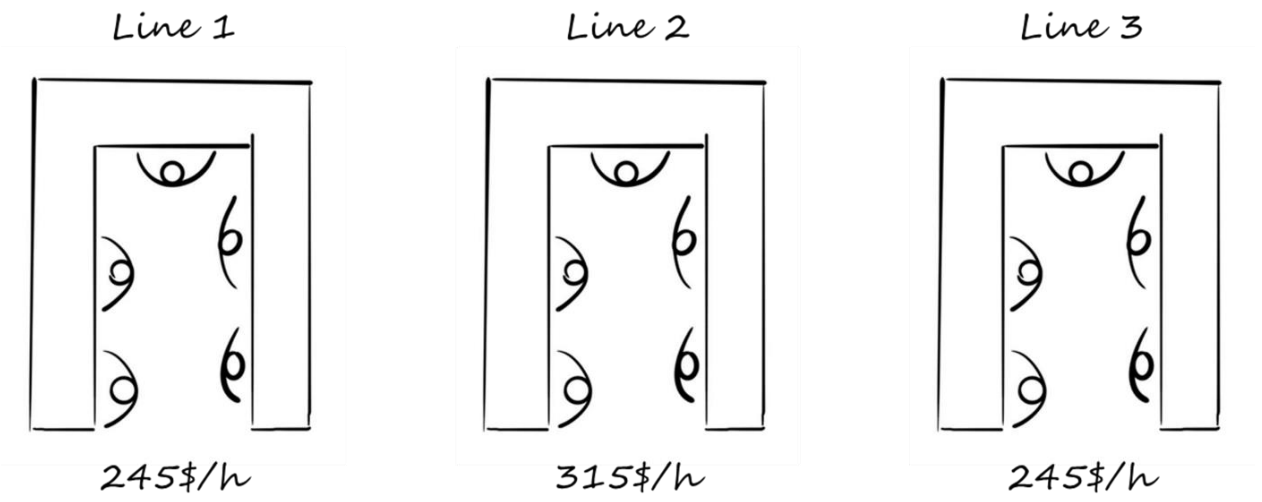
You are the proud owner of a manufacturing company and own three production lines. On each of these lines, you can produce the same products, but unfortunately, each line is designed differently, and the hourly labour cost varies for each line. You want to be able to make the best use of your equipment to be the most cost-efficient.

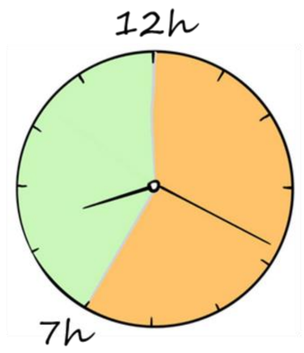
Your factory possesses three production lines. The hourly labour cost is 245$/h for lines 1 and 3 and 315$/h for line 2. Due to the current rules and regulations, you have some constraints on the daily working hours: one line cannot run for less than 7 hours or more than 12 hours per day.

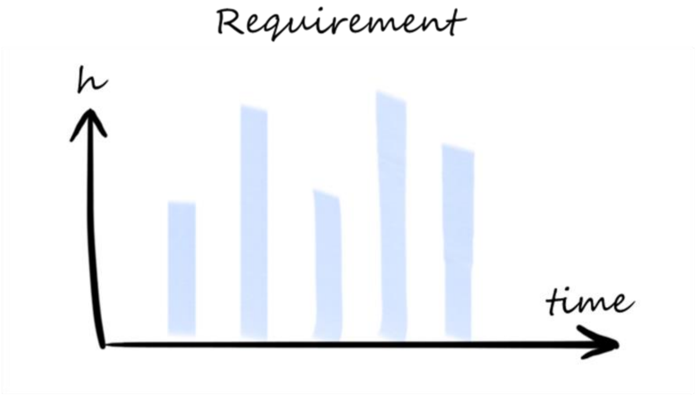
The requirement from your customer is the number of hours of production for each day of a week, and you want to schedule these hours on the day it is required, meaning no early or late planning.

Source: https://towardsdatascience.com/optimization-of-a-weekly-production-plan-with-python-and-gurobi-part-1-d1257ad29a9

Part 1.1 - Workshop Problem







Source: https://towardsdatascience.com/optimization-of-a-weekly-production-plan-with-python-and-gurobi-part-1-d1257ad29a9

Part 1.1 - Workshop Problem

Daily Requirement	Hours/Day
2020/7/13	30
2020/7/14	10
2020/7/15	34
2020/7/16	25
2020/7/17	23
2020/7/18	24
2020/7/19	25

Source: https://towardsdatascience.com/optimization-of-a-weekly-production-plan-with-python-and-gurobi-part-1-d1257ad29a9

Part 1.1 - Workshop Problem

Requirement

$$For\ each\ day:\quad requirementent_{date} = \sum_{workcenter=Line1}^{Line3} total\_hours_{date,workcenter}$$

Objective Function

$$Min\left(\sum_{timeline}\sum_{workcenters} labor\_cost_{date,workcenter}\right)$$

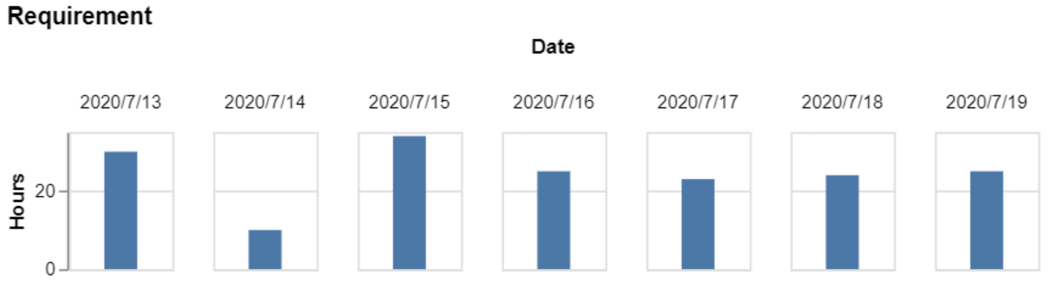
Source: https://towardsdatascience.com/optimization-of-a-weekly-production-plan-with-python-and-gurobi-part-1-d1257ad29a9

Part 1.1 - Workshop Problem

Let’ use Python & Gurobipy

To Solve the Problem

Part 1.1 - Workshop Problem



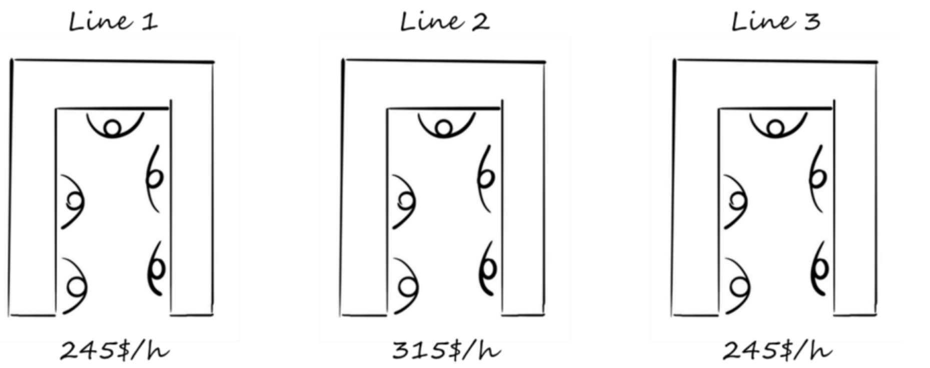

```
Solution count 1: 44065

Optimal solution found (tolerance 1.00e-04)
Best objective 4.406500000000e+04, best bound 4.406500000000e+04, gap 0.0000%
Total cost = $44065.0
```

Part 1.2 - OT and Weekend

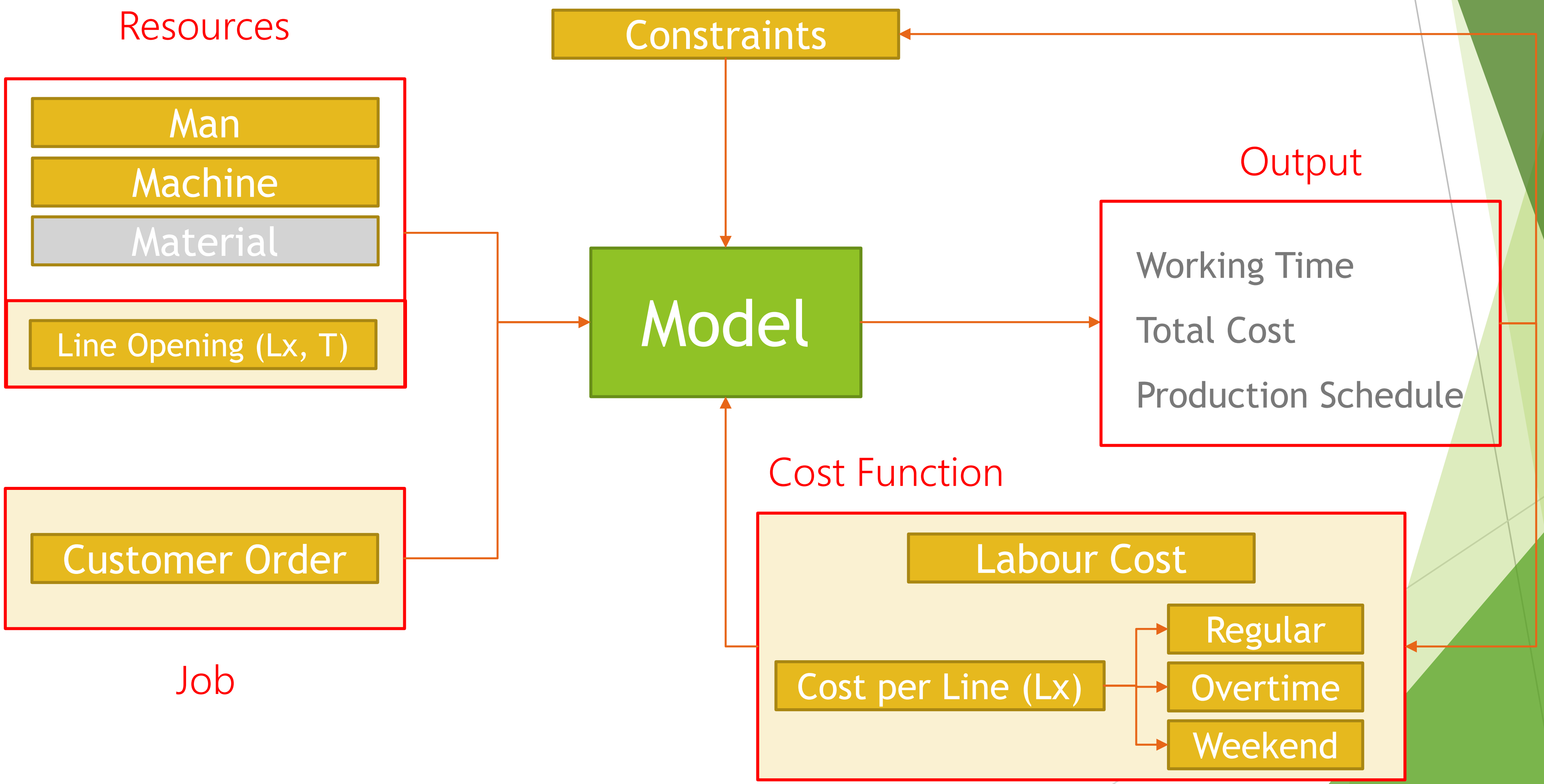
Now, we will see how we can complexify the problem to make it more realistic. Indeed, the model we developed in the first article is limited because we did not consider extra labour costs during weekends and overtime hours

Assuming that up to 8 hours per day our operators are paid at a regular rate whereas overtime hours are paid 50% higher. Also, while working during the weekend, the operators are paid twice the regular rate.



Source: https://towardsdatascience.com/optimization-of-a-weekly-production-plan-with-python-and-gurobi-part2-6ebcefe85e52

Part 1.2 - OT and Weekend

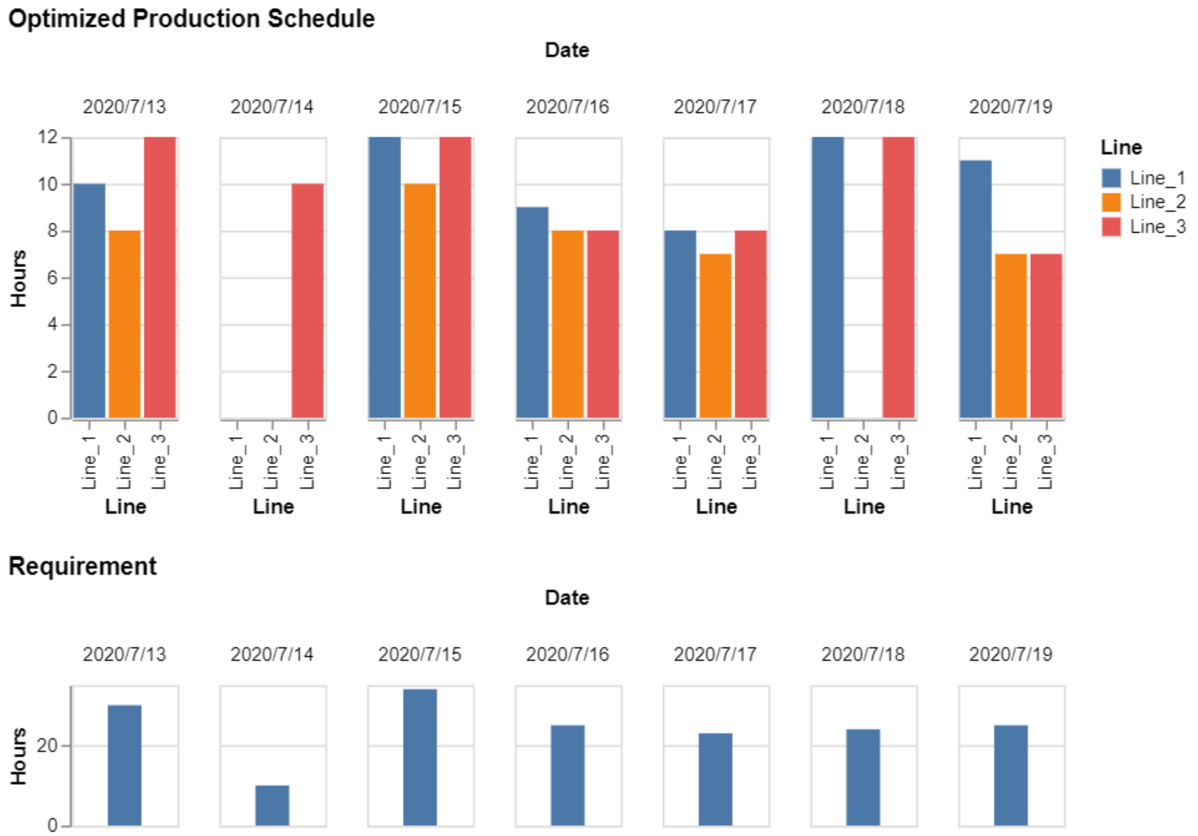


Part 1.2 - OT and Weekend

Let’ use Python & Gurobipy

To Solve the Problem

Part 1.2 - OT and Weekend




```
Solution count 3: 59587.5 59815 59832.5

Optimal solution found (tolerance 1.00e-04)
Best objective 5.958750000000e+04, best bound 5.958750000000e+04, gap 0.0000%
Total cost = $59587.50000000001
```

Part 1.3 - Early Production & Inventory Cost

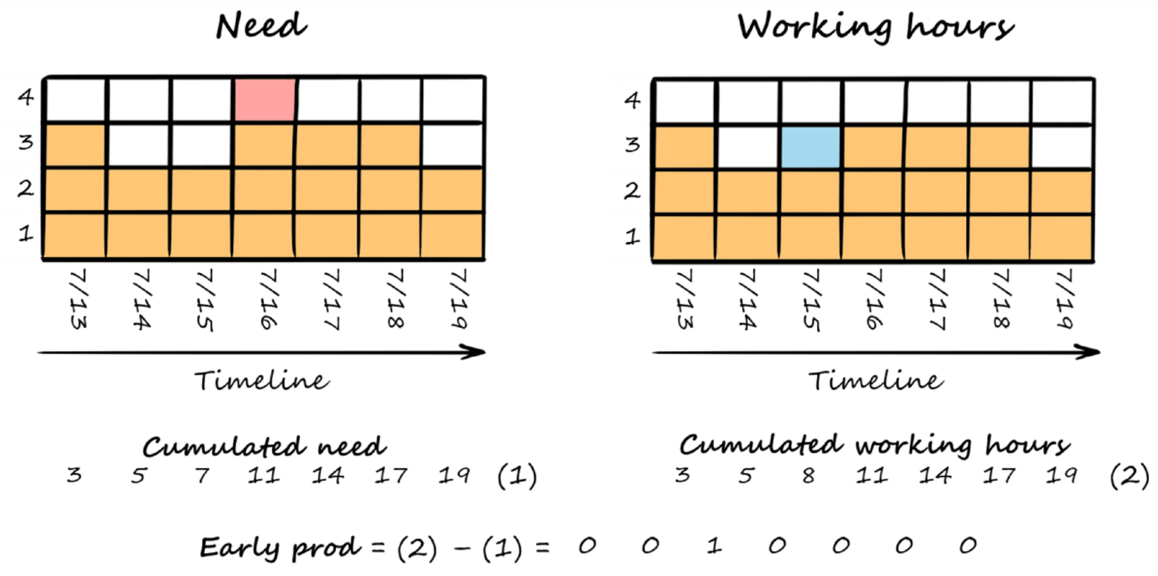
Even though the previous models are very satisfying, they are subject to a major limitation! It is not possible to plan manufacturing on a day different day the required delivery date, possibly leading to many unsolvable situations.

Most of the inputs remain the same, however, a new direct cost is generated: the carrying costs. Carrying costs include all the expenses generated to an organization due to the inventory carried. It includes capital cost, storage costs, and risk costs. Actual carrying costs can vary a lot, depending on your location, the type of products you produce, and many other factors. However, according to APICS, it is often estimated between 20% and 30% for manufacturing industries.

Source: https://towardsdatascience.com/optimization-of-a-weekly-production-plan-with-python-and-gurobi-part2-6ebcefe85e52

Part 1.3 - Early Production & Inventory Cost

In our problem, we are working with hours of production. The unitary carrying costs we need to use is an estimation of the carrying cost of one hour of production stored for one day. We can consider that it is equal to 20–30% of the average inventory produced in one hour. Let’s set it to 25$ for our example.



Source: https://towardsdatascience.com/optimization-of-a-weekly-production-plan-with-python-and-gurobi-part2-6ebcefe85e52

Part 1.3 - Early Production & Inventory Cost

$$Cumulated\ need[k] = \sum_{date=1}^{date=k} needs_{date} \quad (1)$$

$$Cumulated\ working\ hours[k] = \sum_{wc=line\ 1}^{wc=line\ 3} \sum_{date=1}^{date=k} total\_working\_hours_{date,wc} \quad (2)$$

$$Early\ production[k] = \sum_{wc=line\ 1}^{wc=line\ 3} \sum_{date=1}^{date=k} total\_working\_hours_{wc,date} - \sum_{date=1}^{date=k} needs_{date}$$

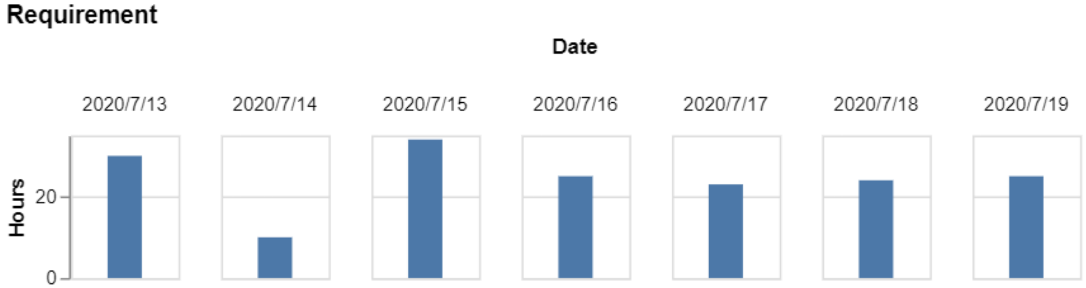
Source: https://towardsdatascience.com/optimization-of-a-weekly-production-plan-with-python-and-gurobi-part2-6ebcefe85e52

Part 1.3 - Early Production & Inventory Cost

Let' use Python & Gurobipy

To Solve the Problem

Part 1.3 - Early Production & Inventory Cost

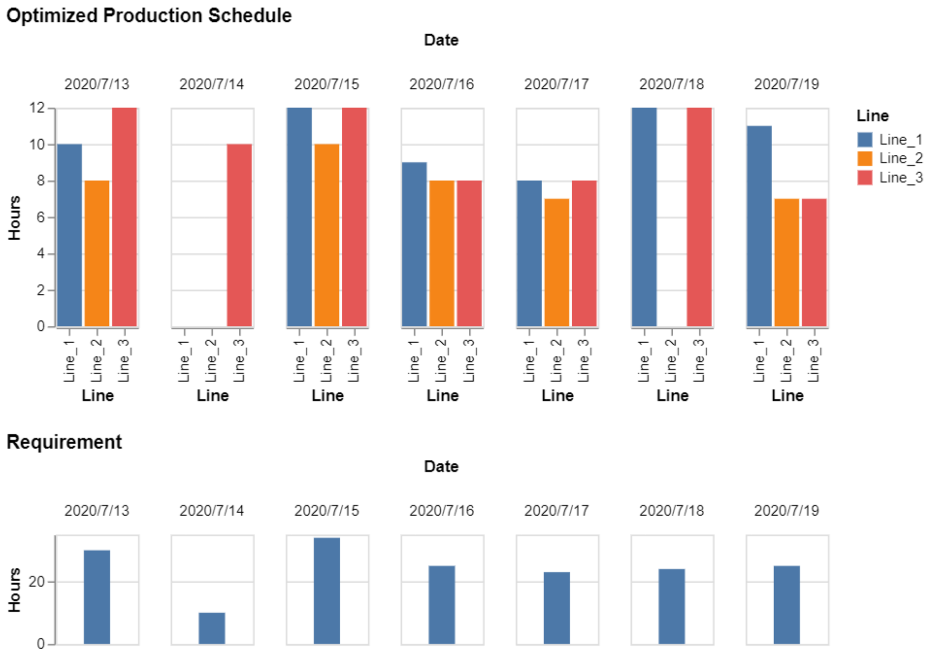



```
Solution count 7: 55440 55557.5 55882.5 ... 62760

Optimal solution found (tolerance 1.00e-04)
Best objective 5.544000000000e+04, best bound 5.544000000000e+04, gap 0.0000%
Total cost = $55440.0
```

Cost Comparison in non-early-production and early-production plan

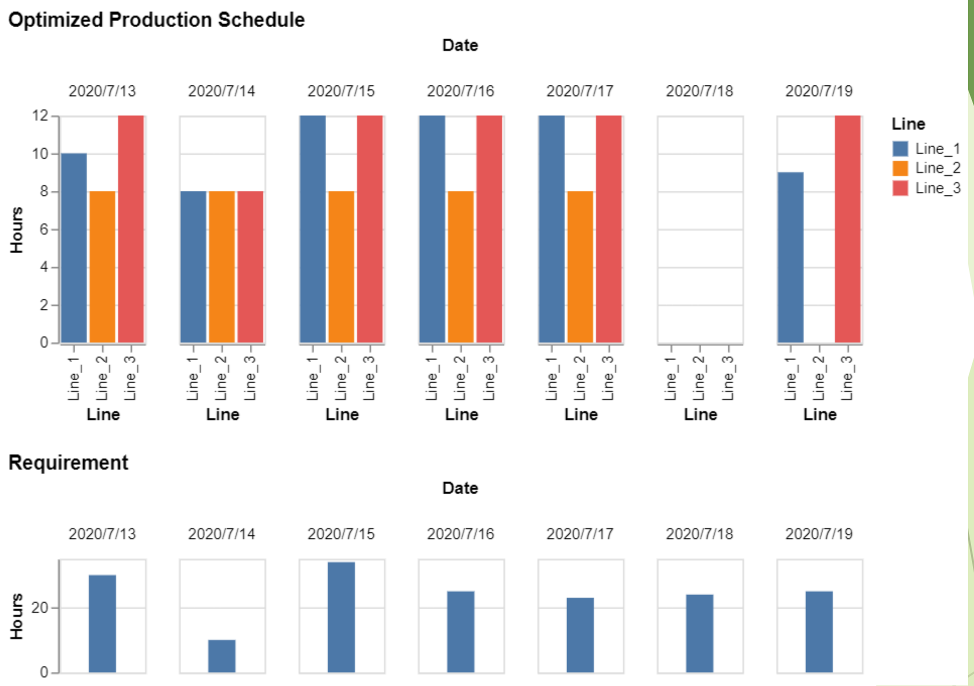
Part 1.2 – Non-early-production




```
Solution count 3: 59587.5 59815 59832.5

Optimal solution found (tolerance 1.00e-04)
Best objective 5.958750000000e+04, best bound 5.95875000000
Total cost = $59587.50000000001
```

Part 1.3 – Early-production with inventory cost




```
Solution count 7: 55440 55557.5 55882.5 ... 62760

Optimal solution found (tolerance 1.00e-04)
Best objective 5.544000000000e+04, best bound 5.54400000
Total cost = $55440.0
```