INDR371 - HW7

Umur Berkay Karakaş

January 14, 2023

Question 1

- x_k: the number of product k produced per minute

$$
\begin{aligned}
\max \quad & x_a && \text{(1a)}\\
\text{s.t.} \quad & 5x_a + 5x_b + 5x_c \leq 5 && \text{(1b)}\\
& 3x_a + 4x_b + 5x_c \leq 4 && \text{(1c)}\\
& 15x_a + 0x_b + 0x_c \leq 4 && \text{(1d)}\\
& 0x_a + 4x_b + 3x_c \leq 6 && \text{(1e)}\\
& 10x_a + 10x_b + 10x_c \leq 8 && \text{(1f)}\\
& 5x_a - 7x_b + 0x_c = 0 && \text{(1g)}\\
& 0x_a + 4x_b - 5x_c = 0 && \text{(1h)}\\
& x_a, x_b, x_c \in \mathbf{R} && \text{(1i)}
\end{aligned}
$$

- Constraints 1b, 1c, 1d, 1e and 1f are for the capacity limitations for resource 1, 2, 3, 4 and 5.
- Constraints 1g and 1h are for the batch demand requirement, which is 7 units of A, 5 units of B and 4 units of C.

Question 1

In [2]:
```
activity_times = np.array([[5,5,5], [3,4,5], [15,0,0], [0,4,3], [10,10,10]])
workers = [5,4,4,6,8]
products = ["a", "b", "c"]
demands = [7, 5, 4]
```

In [3]:
```
m = gp.Model("q1")
m.ModelSense = GRB.MAXIMIZE
x = m.addVars(products, vtype=GRB.CONTINUOUS)
for i in range(len(activity_times)):
    m.addConstr(gp.quicksum(activity_times[i][j]*x[products[j]]
                            for j in range(len(activity_times[i]))) <= workers[i])
m.addConstr(demands[1]*x["a"] == demands[0]*x["b"])
m.addConstr(demands[2]*x["b"] == demands[1]*x["c"])
m.setObjective(x["a"])
```

```
Set parameter Username
Academic license - for non-commercial use only - expires 2023-10-08
```

Figure 1: LP model implementation for the model in Question 1

```
In [10]: used_workers = np.matmul(activity_times,[x[a].X for a in x.keys()])
         bottleneck_resource = np.where(workers==used_workers)[0][0] + 1
         print(bottleneck_resource)

         3

In [11]: max_flow_rates = 480 * np.array([_.x for _ in x.values()])
         max_flow_rates

Out[11]: array([128.        ,  91.42857143,  73.14285714])

In [12]: utilizations = used_workers/workers
         utilizations

Out[12]: array([0.60952381, 0.58095238, 1.        , 0.2031746 , 0.76190476])
```

Figure 2: LP model results for the model in Question 1

From Figure 2, we can infer the following:

- **Bottleneck resource** is the resource 3.
- **Maximum daily flow rates** for products A, B and C are 128, 91.43 and 73.14.
- **Utilizations of resources 1, 2, 3, 4 and 5** are 0.61, 0.58, 1, 0.20 and 0.76.

Question 2

- x_k: the number of product k produced per minute
- i_k: the number of inexperienced workers allocated to resource k
- e_k: the number of experienced workers allocated to resource k

$$\begin{aligned}
\max \quad & x_a && \text{(2a)}\\
\text{s.t.} \quad & 5x_a + 5x_b + 5x_c \leq e_1 + 2/3i_1 && \text{(2b)}\\
& 3x_a + 4x_b + 5x_c \leq e_2 + 2/3i_2 && \text{(2c)}\\
& 15x_a + 0x_b + 0x_c \leq e_3 + 2/3i_3 && \text{(2d)}\\
& 0x_a + 4x_b + 3x_c \leq e_4 + 2/3i_4 && \text{(2e)}\\
& 10x_a + 10x_b + 10x_c \leq e_5 + 2/3i_5 && \text{(2f)}\\
& 5x_a - 7x_b + 0x_c = 0 && \text{(2g)}\\
& 0x_a + 4x_b - 5x_c = 0 && \text{(2h)}\\
& \sum_{i=1}^{5} e_i = 12 && \text{(2i)}\\
& \sum_{i=1}^{5} i_i = 15 && \text{(2j)}\\
& x_a, x_b, x_c \in \mathbf{R} && \text{(2k)}\\
& e_k, i_k \in \mathbf{Z} \qquad \forall k && \text{(2l)}
\end{aligned}$$

- Constraints 2b, 2c, 2d, 2e and 2f are for the capacity limitations for resource 1, 2, 3, 4 and 5.
- Constraints 2g and 2h are for the batch demand requirement, which is 7 units of A, 5 units of B and 4 units of B.
- Constraints 2i and 2j are for the number of experienced and inexperienced workers.

In this model, I simply changed the capacity of each resource from a fixed number as in Question 1 to a sum of 2 decision variables which correspond to experienced and inexperienced workers. I multiplied the decision variable for inexperienced workers in each constraint by 2/3 because time requirement of an inexperienced worker is 1.5 times of an experienced worker.

Question 2

In [39]:
```
exp_times = activity_times
inexp_times = 1.5 * exp_times
workers = [12, 15]
```

In [40]:
```
m = gp.Model("q2")
m.ModelSense = GRB.MAXIMIZE
x = m.addVars(products, vtype=GRB.CONTINUOUS)
inexp = m.addVars(range(5), vtype=GRB.INTEGER)
exp = m.addVars(range(5), vtype=GRB.INTEGER)
for i in range(len(exp_times)):
    m.addConstr(gp.quicksum(exp_times[i][j]*x[products[j]]
                            for j in range(len(exp_times[i]))) <= exp[i]+2*inexp[i]/3)
m.addConstr(gp.quicksum(exp[_] for _ in range(5)) == workers[0])
m.addConstr(gp.quicksum(inexp[_] for _ in range(5)) == workers[1])
m.addConstr(demands[1]*x["a"] == demands[0]*x["b"])
m.addConstr(demands[2]*x["b"] == demands[1]*x["c"])
m.setObjective(x["a"])
```

Figure 3: LP model implementation for the model in Question 2

In [43]:
```
inexp
```

Out[43]:
```
{0: <gurobi.Var C3 (value -0.0)>,
 1: <gurobi.Var C4 (value -0.0)>,
 2: <gurobi.Var C5 (value 2.0)>,
 3: <gurobi.Var C6 (value 1.0)>,
 4: <gurobi.Var C7 (value 12.0)>}
```

In [44]:
```
exp
```

Out[44]:
```
{0: <gurobi.Var C8 (value 4.0)>,
 1: <gurobi.Var C9 (value 3.0)>,
 2: <gurobi.Var C10 (value 4.0)>,
 3: <gurobi.Var C11 (value 1.0)>,
 4: <gurobi.Var C12 (value 0.0)>}
```

Figure 4: LP model results for the model in Question 2

From Figure 4 we can infer the following table:

Resource	Inexperienced Workers	Experienced Workers
1	0	4
2	0	3
3	2	4
4	1	1
5	12	0

Table 1: Worker allocation to the resources