IEOR E4004 HW2 Mark Ma km4054

Question 1.

1、Variables:

① $x_i$, $i \in \{1,2,3,4,5,6\}$: Indicates the number of air traffic controllers who work 8-hour shifts, starting at 12 am, 4am, 8am, 12 pm, 4pm, 8pm respectively.

② $y_j$, $j \in \{1,2,3,4\}$: Indicates the number of air traffic controllers who work 12-hour shifts, starting at 12am, 8am, 12pm, 8pm respectively.

Objective: min $40 \times 8 \times \sum_{i=1}^{6} x_i + 35 \times 12 \sum_{j=1}^{4} y_j$

Constraints:

① Time Slot constraints:

1) 12 am to 4pm: $x_1 + x_6 + y_1 + y_4 \geq 8$

2) 4am to 8 am: $x_1 + x_2 + y_1 + y_4 \geq 10$

3) 8am to 12pm: $x_2 + x_3 + y_1 + y_2 \geq 16$

4) 12pm to 4pm: $x_3 + x_4 + y_2 + y_3 \geq 21$

5) 4pm to 8pm: $x_4 + x_5 + y_2 + y_3 \geq 18$

6) 8pm to 12am: $x_5 + x_6 + y_3 + y_4 \geq 12$

② Non-negativity and Integer constraints:

$x_i \geq 0$, $x_i \in \mathbb{Z}$; $\forall i \in \{1,2,3,4,5,6\}$

$y_j \geq 0$, $y_j \in \mathbb{Z}$; $\forall j \in \{1,2,3,4\}$

As a result, the algebraic formulation of the problem should be:

$$\min \ 40 \times 8 \times \sum_{i=1}^{6} x_i + 35 \times 12 \sum_{j=1}^{4} y_j$$

s.t.

$$x_1 + x_6 + y_1 + y_4 \geq 8$$

$$x_1 + x_2 + y_1 + y_4 \geq 10$$

$$x_2 + x_3 + y_1 + y_2 \geq 16$$

$$x_3 + x_4 + y_2 + y_3 \geq 21$$

$$x_4 + x_5 + y_2 + y_3 \geq 18$$

$$x_5 + x_6 + y_3 + y_4 \geq 12$$

$$x_i \geq 0 \;, \; x_i \in \mathbb{Z} \quad \forall i \in \{1, 2, 3, 4, 5, 6\}$$

$$y_j \geq 0 \;, \; y_j \in \mathbb{Z} \quad \forall j \in \{1, 2, 3, 4\}$$

Below is the result of using Gurobi to solve the problem, the codes will be attached as an appendix.

```
if model.status == gp.GRB.OPTIMAL:
    print("Optimal solution found:")
    print(f"x (8-hour shift): {[x[i].x for i in range(6)]}")
    print(f"y (12-hour shift): {[y[j].x for j in range(4)]}")
    print(f"Total labor cost: ${model.objVal:.2f}")
else:
    print("No optimal solution found.")
[10]  ✓ 0.0s
...
Optimal solution found:
x (8-hour shift): [-0.0, 2.0, 3.0, 3.0, -0.0, -0.0]
y (12-hour shift): [8.0, 3.0, 12.0, -0.0]
Total labor cost: $12220.00
```

The minimized dispatcher labor cost is \$12220, with

$x = [0, 2, 3, 3, 0, 0]$

$y = [8, 3, 12, 0]$

2. We add an additional constraint to the algebraic formulation above:

$$\frac{\sum_{j=1}^{4} y_j}{\sum_{i=1}^{6} x_i + \sum_{j=1}^{4} y_j} \leq \frac{1}{3} \iff \underline{2\sum_{j=1}^{4} y_j \leq \sum_{i=1}^{6} x_i}$$

Below is the result of using Gurobi to solve the new problem, codes will be attached as an appendix.

```
if model.status == gp.GRB.OPTIMAL:
    print("Optimal solution found:")
    print(f"x (8-hour shift): {[x[i].x for i in range(6)]}")
    print(f"y (12-hour shift): {[y[j].x for j in range(4)]}")
    print(f"Total labor cost: ${model.objVal:.2f}")
else:
    print("No optimal solution found.")
[16]  ✓ 0.0s
...
Optimal solution found:
x (8-hour shift): [2.428571428571428, 2.0, 8.428571428571429, 6.0, 5.428571428571429, 0.0]
y (12-hour shift): [5.571428571428572, 0.0, 6.571428571428571, 0.0]
Total labor cost: $12871.43
```

The minimized dispatcher labor cost is now \$12871.43, with

$x = [2.43, 2, 8.43, 6, 5.43, 0]$

$y = [5.57, 0, 6.57, 0]$

## Question 2

The primal problem is:

$$\max \; 10x_1 + 14x_2 + 20x_3$$

$$\text{s.t.} \quad 2x_1 + 3x_2 + 4x_3 \leq 220$$

$$4x_1 + 2x_2 - x_3 \leq 385$$

$$x_1 + 4x_3 \leq 160$$

$$x_1, x_2, x_3 \geq 0$$

The dual problem is:

$$\min \; 220y_1 + 385y_2 + 160y_3$$

$$\text{s.t.} \quad 2y_1 + 4y_2 + y_3 \geq 10$$

$$3y_1 + 2y_2 \geq 14$$

$$4y_1 - y_2 + 4y_3 \geq 20$$

$$y_1, y_2, y_3 \geq 0$$

Firstly, we solve the primal using Gurobi, the result is shown below, and the codes are attached as appendix.

```
if model.status == gp.GRB.OPTIMAL:
    print("Optimal solution found:")
    print(f"x: {[x[i].x for i in range(3)]}")
    print(f"Maximized Value: ${model.objVal:.2f}")
else:
    print("No optimal solution found.")
✓ 0.0s

Optimal solution found:
x: [97.77777777777777, 0.0, 6.111111111111114]
Maximized Value:  1100.00
```

The maximum of primal should be 1100, with corresponding $x = [97.77, 0, 6.11]$

Then, we solve the dual using Gurobi, the result is shown below, and the codes are attached as appendix.

```
if model.status == gp.GRB.OPTIMAL:
    print("Optimal solution found:")
    print(f"y: {[y[i].x for i in range(3)]}")
    print(f"Minimized Value: ${model.objVal:.2f}")
else:
    print("No optimal solution found.")
✓ 0.0s

Optimal solution found:
y: [5.0, 0.0, 0.0]
Minimized Value:  1100.00
```

The minimum of dual should be 1100, with corresponding $y = [5, 0, 0]$.

Therefore, the primal and dual indeed yield the same optimal value, which is 1100.

## Question 3

1. All sensible patterns for 10-ft cutting are listed as follows:

| Pattern # | Pattern Combination | Scrap pieces leftover |
|---|---|---|
| 1 $(x_1)$ | 3+3+3 | 1 |
| 2 $(x_2)$ | 3+3+4 | 0 |
| 3 $(x_3)$ | 4+4 | 2 |
| 4 $(x_4)$ | 3+5 | 2 |
| 5 $(x_5)$ | 4+5 | 1 |
| 6 $(x_6)$ | 5+5 | 0 |

2. (a) Variables: $x_i, \forall i \in \{1,2,3,4,5,6\}$, where $x_i$ represents the 10-ft boards used in each pattern stated above.

Objective: $\min \sum_{i=1}^{6} x_i$

Constraints: $3x_1 + 2x_2 + x_4 \geq 90$

$$x_2 + 2x_3 + x_5 \geq 60$$

$$x_4 + x_5 + 2x_6 \geq 60$$

$$x_i \geq 0, \ x_i \in \mathbb{Z}, \ \forall i = \{1, 2, 3, 4, 5, 6\}$$

Therefore, the algebraic formulation of the problem is as follows:

$$\min \sum_{i=1}^{6} x_i$$

$$\text{s.t.} \quad 3x_1 + 2x_2 + x_4 \geq 90$$

$$x_2 + 2x_3 + x_5 \geq 60$$

$$x_4 + x_5 + 2x_6 \geq 60$$

$$x_i \geq 0, \ x_i \in \mathbb{Z}$$

$$\forall i = \{1, 2, 3, 4, 5, 6\}$$

(b) Below is the result from Gurobi of (a), the code is attached as an appendix.

```
if model.status == gp.GRB.OPTIMAL:
    print("Optimal solution found:")
    print(f"x: {[x[i].x for i in range(6)]}")
    print(f"Minimized Value: {model.objVal:.2f}")
else:
    print("No optimal solution found.")
✓ 0.0s

Optimal solution found:
x: [-0.0, 46.0, 7.0, -0.0, -0.0, 30.0]
Minimized Value: 83.00
```

From the result, the optimal number of the patterns should be $x = [0, 46, 7, 0, 0, 30]$, that is, $x_1 = 0$, $x_2 = 46$, $x_3 = 7$, $x_4 = x_5 = 0$, $x_6 = 30$

The minimum number of 10-ft boards to cut is 83.

However, the optimal solution isn't unique, but they yield the same optimal value, a few examples can be shown below:

```
# Output all the solutions found
solution_count = min(10, model.SolCount)  # Number of solutions in the solution pool
print(f"Number of solutions found: {solution_count}")

if model.Status == gp.GRB.OPTIMAL or model.SolCount > 0:
    print('Optimal objective value: %g' % model.objVal)
    for i in range(min(10,solution_count)):
        model.setParam(gp.GRB.Param.SolutionNumber, i)
        print(f"Solution {i + 1}: {model.PoolObjVal}")
        print(f"x: {[x[j].xn for j in range(6)]}")
✓ 0.0s

Number of solutions found: 10
Optimal objective value: 83
Solution 1: 83.0
x: [0.0, 46.0, 6.0, 0.0, 2.0, 29.0]
Solution 2: 82.99999999999999
x: [-0.0, 45.0, 7.0, -0.0, 1.9999999999999662, 29.000000000000018]
Solution 3: 83.0
x: [-0.0, 45.0, 8.0, -0.0, -0.0, 30.0]
Solution 4: 83.0
x: [-0.0, 46.0, 7.0, -0.0, -0.0, 30.0]
Solution 5: 83.0
x: [-0.0, 45.0, 7.0000000000000036, 1.0, 0.9999999999999929, 29.0]
Solution 6: 83.0
x: [-0.0, 45.0, 7.0000000000000036, -0.0, 0.9999999999999929, 30.0]
Solution 7: 83.0
x: [-0.0, 45.0, 6.0, -0.0, 3.0, 29.0]
Solution 8: 83.0
x: [-0.0, 45.0, 6.0, -0.0, 4.0, 28.0]
Solution 9: 83.0
x: [-0.0, 46.0, 5.0, -0.0, 4.0, 28.0]
Solution 10: 83.0
x: [1.0, 44.0, 7.0, -0.0, 2.0, 29.0]
```

As stated in the picture on the left, there're several optimal solution, all of them have the same model objective value 83.

3. (a) In the chart stated in Q1, we focus on the scrap pieces leftover. Then, we need to minimize $x_1 + 2x_3 + 2x_4 + x_5$. With respect to the above solution, we add a constraint $\sum_{i=1}^{6} x_i \le 83$.

Then, the algebraic formulation becomes:

$$
\begin{aligned}
\min \quad & x_1 + 2x_3 + 2x_4 + x_5 \\
\text{s.t.} \quad & \sum_{i=1}^{6} x_i \le 83 \\
& 3x_1 + 2x_2 + x_4 \ge 90 \\
& x_2 + 2x_3 + x_5 \ge 60 \\
& x_4 + x_5 + 2x_6 \ge 60 \\
& x_i \ge 0, \quad x_i \in \mathbb{Z} \\
& \forall i = \{1, 2, 3, 4, 5, 6\}
\end{aligned}
$$

(b) The model can be modified as above, the result is shown below, the codes are attached as an appendix.

```
# Print the results
if model.status == gp.GRB.OPTIMAL:
    print("Optimal solution found:")
    for i in range(6):
        print(f"Pattern {i+1} (x[{i}]): {x[i].x}")
    print(f"Total number of 10-ft boards: {sum([x[i].x for i in range(6)]):.0f}")
    total_scrap = x[0].x + 2 * x[2].x + 2 * x[3].x + x[4].x
    print(f"Total scrap generated: {total_scrap} inches")
else:
    print("No optimal solution found.")
```

✓ 0.0s

```
Optimal solution found:
Pattern 1 (x[0]): -0.0
Pattern 2 (x[1]): 46.0
Pattern 3 (x[2]): 7.0
Pattern 4 (x[3]): -0.0
Pattern 5 (x[4]): -0.0
Pattern 6 (x[5]): 30.0
Total number of 10-ft boards: 83
Total scrap generated: 14.0 inches
```

Then, $x_1 = x_4 = x_5 = 0$, $x_2 = 46$, $x_3 = 7$, $x_6 = 30$. The minimum pieces leftover is 14, the total number of boards used is still 83.