DDA 5002 Optimization Fall 2025

Homework # 1

Student ID: 225040065

Question 1

(a)

x_1 : number of the product of the first type produced per day

x_2 : number of the product of the second type produced per day

objective function:

$$\max(7.8x_1 + 7.1x_2)$$

constraints:

$$\frac{1}{4}x_1 + \frac{1}{3}x_2 \leq 90$$
$$\frac{1}{8}x_1 + \frac{1}{3}x_2 \leq 80$$
$$x_1, x_2 \geq 0$$

(b)

(i) **can** be easily incorporated into the linear program formulation, while (ii) **cannot**.

for (i) modification (Linear Program Formulation)

x_1 : number of the product of the first type produced per day

x_2 : number of the product of the second type produced per day

h : number of hours of overtime assembly labor per day

objective function:

$$\begin{aligned} &\max(7.8x_1 + 7.1x_2 - 7h) \\ s.t. \quad &\frac{1}{4}x_1 + \frac{1}{3}x_2 \leq 90 + h \\ &\frac{1}{8}x_1 + \frac{1}{3}x_2 \leq 80 \\ &h \leq 50 \\ &x_1, x_2, h \geq 0 \\ &x_1, x_2, h \in \mathbb{Z} \end{aligned}$$

for (ii) modification

x_1 : number of the product of the first type produced per day

x_2 : number of the product of the second type produced per day

objective function:

$$\max(7.8x_1 + 7.1x_2 - \mathbb{1}\{7.8x_1 + 7.1x_2 > 300\}(0.12x_1 + 0.09x_2))$$

constraints:

$$\begin{gathered}\frac{1}{4}x_1 + \frac{1}{3}x_2 \le 90 \\ \frac{1}{8}x_1 + \frac{1}{3}x_2 \le 80 \\ x_1, x_2 \ge 0 \\ x_1, x_2, h \in \mathbb{Z}\end{gathered}$$

Question 2

(a)

p_k : daily power consume of the k-th car, and we have: $p_1 = 10, p_2 = 8, p_3 = 13, p_4 = 15, p_5 = 9$

d_k : total mileage traveled of the k-th car, and we have: $d_1 = 60, d_2 = 55, d_3 = 75, d_4 = 80, d_5 = 64$

m, b : linear program parameters

objective function:

$$\min(\Sigma_{k=1}^{5} y_k)$$

constraints:

$$\begin{gathered}-y_k \le d_k - (mp_k + b) \le y_k, \quad k = 1,2,3,4,5 \\ m, b, y_k \ge 0, \quad k = 1,2,3,4,5\end{gathered}$$

(b)

code and output:

```
import coptpy as cp
from coptpy import COPT

# Create COPT environment
env = cp.Envr()

# Create COPT model
model = env.createModel("q2")

# Add variables
m = model.addVar(lb=0, name="m")
b = model.addVar(lb=0, name="b")
y = [model.addVar(lb=0, name=f"y_{i}") for i in range(5)]

p = [10, 8, 13, 15, 9]
d = [60, 55, 75, 80, 64]

# Add constraints
for i in range(5):
    model.addConstr(d[i] - (m * p[i] + b) <= y[i])
    model.addConstr(d[i] - (m * p[i] + b) >= -y[i])
```

```python
# Set objective function
model.setObjective(sum(y), sense=COPT.MINIMIZE)

# Set parameter
model.setParam(COPT.Param.TimeLimit, 10.0)

# Solve the model
model.solve()

# Analyze solution
if model.status == COPT.OPTIMAL:
    print(f"Objective value: {model.objval:.6f}")
    print(f"Variable solution: m = {m.x:.4f}, b = {b.x:.4f}")
```

```
Minimizing an LP problem

The original problem has:
    10 rows, 7 columns and 30 non-zero elements
The presolved problem has:
    10 rows, 7 columns and 30 non-zero elements

Starting the simplex solver using up to 8 threads

Problem info:
Range of matrix coefficients:    [1e+00,2e+00]
Range of rhs coefficients:       [6e+01,8e+01]
Range of bound coefficients:     [0e+00,0e+00]
Range of cost coefficients:      [1e+00,1e+00]

Method   Iteration           Objective  Primal.NInf   Dual.NInf        Time
Dual             0    0.0000000000e+00            5           0       0.00s
Dual             6    9.7158255024e+00            0           0       0.00s
Postsolving
Dual             6    9.7142857143e+00            0           0       0.00s

Solving finished
Status: Optimal  Objective: 9.7142857143e+00  Iterations: 6  Time: 0.00s
Objective value: 9.714286
Variable solution: m = 3.5714, b = 26.4286
```

report:

Based on the data, we found that for every 1 unit increase in power consumption, the driving distance increases by an average of 4.5 miles. This model can fit the observed data well because it minimizes the sum of absolute errors between the predicted and actual values.

Question 3

period: $t = 1, 2, \cdots, T$

plant: $p = 1, 2, \cdots, P$

outlet: $o = 1, 2, \cdots, O$

$x_{t,p}$: number of plant p manufatured in period t

$m_{t,p}$: cost of manufaturing plant p in period t

$y_{t,p,o}$: number of plant p shipped to outlet o in period t

$c_{p,o}$: cost of shipping plant p to outlet o in period t

$z_{t,o}$: number of plant sold to outlet o in period t

$r_{t,o}$: price of selling plant to outlet o in period t

$s_{t,p}$: number of plant p stored in period t

h_p : cost of storing plant p in period t

$X_{t,p}$: the max number of plant p manufatured in period t

$Z_{t,o}$: the max number of plant p sold to outlet o

S : the max number of plant stored

objective function:

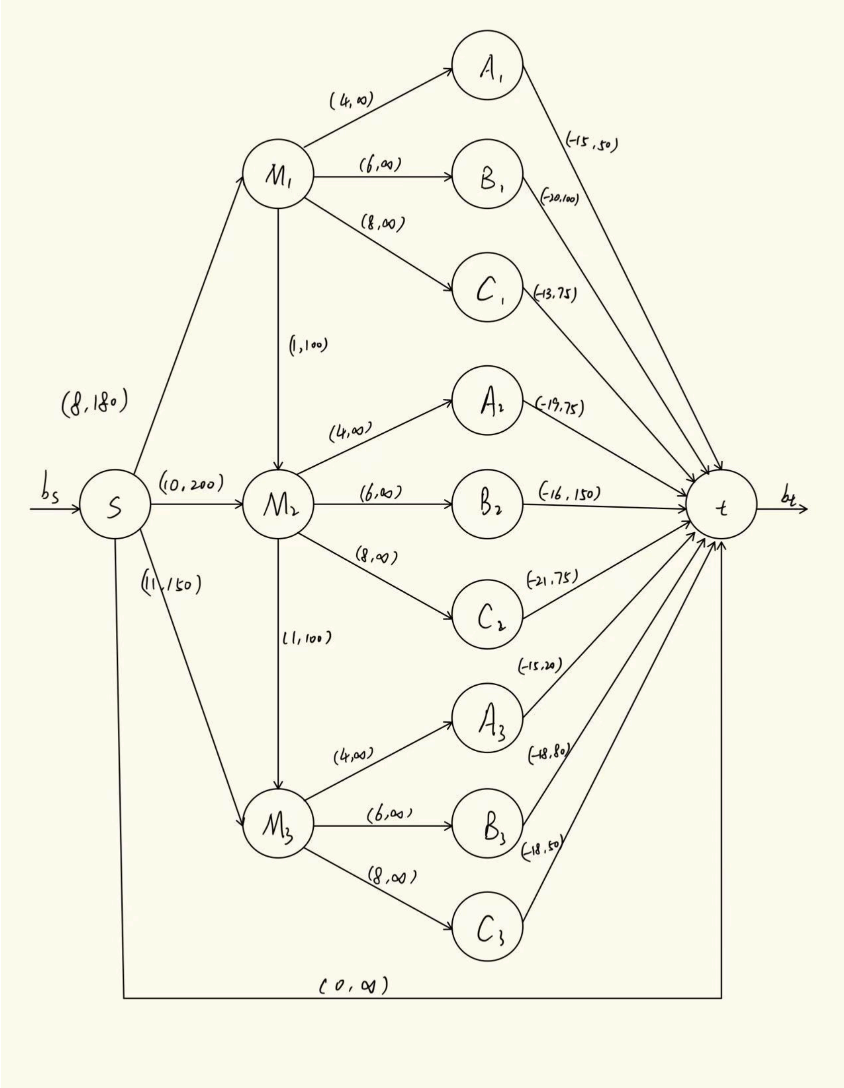
$$\min(\Sigma_{t=1}^{T}\Sigma_{o=1}^{O}r_{t,o}z_{t,o} - \Sigma_{t=1}^{T}\Sigma_{p=1}^{P}m_{t,p}x_{t,p} - \Sigma_{t=1}^{T}\Sigma_{p=1}^{P}\Sigma_{o=1}^{O}c_{p,o}y_{t,p,o} - \Sigma_{t=1}^{T}\Sigma_{p=1}^{P}h_p s_{t,p})$$

constraints:

$$\begin{gathered} x_{t,p} + s_{t-1,p} = \Sigma_{o=1}^{O}y_{t,p,o} + s_{t,p} \\ \Sigma_{p=1}^{P}y_{t,p,o} \geq z_{t,o} \\ x_{t,p} \leq X_{t,p} \\ z_{t,o} \leq Z_{t,o} \\ s_{t,p} \leq S \\ x_{t,p}, y_{t,p,o}, z_{t,o}, s_{t,p} \geq 0 \\ x_{t,p}, y_{t,p,o}, z_{t,o}, s_{t,p} \in \mathbb{Z} \end{gathered}$$

Question 4

(a)



(b)

as the diagram above, for each edge$_{i,j}$ we have $(c_{i,j}, u_{i,j})$

$x_{i,j}$: number of the units from i to j

$c_{i,j}$: cost of the flow from i to j per unit

$u_{i,j}$: max units of the flow from i to j

b_i : supply for node i

objective function:

$$\min(\Sigma_{i,j\in A} c_{i,j} x_{i,j})$$

constraints:

$$\Sigma_{k\in IN(i)} x_{k,i} + b_i = \Sigma_{j\in OUT(i)} x_{i,j}, \quad \forall i \in A$$

$$0 \le x_{i,j} \le u_{i,j}, \quad \forall i,j \in A$$

$$b_i = \begin{cases} b_s, & i = s, \\ -b_s, & i = t, \\ 0, & i \ne s,t. \end{cases}$$

(c)

codes:

```
import coptpy as cp
from coptpy import COPT

# Create COPT environment
env = cp.Envr()

# Create COPT model
model = env.createModel("q2")

# Add variables
vertex = ["s", "m1", "m2", "m3", "a1", "b1", "c1", "a2", "b2", "c2", "a3", "b3", "c3","t"]
edge = dict()
for v in vertex:
    edge[v] = dict()
edge["s"]["m1"] = [8, model.addVar(lb=0, ub=180, name="s_m1")]
edge["s"]["m2"] = [10, model.addVar(lb=0, ub=200, name="s_m2")]
edge["s"]["m3"] = [11, model.addVar(lb=0, ub=150, name="s_m3")]
edge["m1"]["a1"] = [4, model.addVar(lb=0, name="m1_a1")]
edge["m1"]["b1"] = [6, model.addVar(lb=0, name="m1_b1")]
edge["m1"]["c1"] = [8, model.addVar(lb=0, name="m1_c1")]
edge["m1"]["m2"] = [1, model.addVar(lb=0, ub=100, name="m1_m2")]
edge["m2"]["a2"] = [4, model.addVar(lb=0, name="m2_a2")]
edge["m2"]["b2"] = [6, model.addVar(lb=0, name="m2_b2")]
edge["m2"]["c2"] = [8, model.addVar(lb=0, name="m2_c2")]
edge["m2"]["m3"] = [1, model.addVar(lb=0, ub=100, name="m2_m3")]
edge["m3"]["a3"] = [4, model.addVar(lb=0, name="m3_a3")]
edge["m3"]["b3"] = [6, model.addVar(lb=0, name="m3_b3")]
edge["m3"]["c3"] = [8, model.addVar(lb=0, name="m3_c3")]
edge["a1"]["t"] = [-15, model.addVar(lb=0, ub=50, name="a1_t")]
edge["b1"]["t"] = [-20, model.addVar(lb=0, ub=100, name="b1_t")]
edge["c1"]["t"] = [-13, model.addVar(lb=0, ub=75, name="c1_t")]
edge["a2"]["t"] = [-19, model.addVar(lb=0, ub=75, name="a2_t")]
edge["b2"]["t"] = [-16, model.addVar(lb=0, ub=150, name="b2_t")]
edge["c2"]["t"] = [-21, model.addVar(lb=0, ub=75, name="c2_t")]
edge["a3"]["t"] = [-15, model.addVar(lb=0, ub=20, name="a3_t")]
edge["b3"]["t"] = [-18, model.addVar(lb=0, ub=80, name="b3_t")]
edge["c3"]["t"] = [-18, model.addVar(lb=0, ub=50, name="c3_t")]
edge["s"]["t"] = [0, model.addVar(lb=0, name="s_t")]

bs = model.addVar(lb=0, ub=530, name="bs")

# Add constraints
for v in vertex:
    model.addConstr(sum(edge[v][j][1] for j in vertex if j in edge[v]) - sum(edge[i][v][1]
for i in vertex if v in edge[i]) == (bs if v == "s" else -bs if v == "t" else 0))

# Set objective function
model.setObjective(sum(edge[v1][v2][0] * edge[v1][v2][1] for v1 in vertex for v2 in vertex
if v2 in edge[v1]), sense=COPT.MINIMIZE)

# Set parameter
```

```python
model.setParam(COPT.Param.TimeLimit, 10.0)

# Solve the model
model.solve()

# Analyze solution
if model.status == COPT.OPTIMAL:
    print(f"Objective value: {model.objval:.6f}")
    allvars = model.getVars()
    print("Variable solution:")
    for var in allvars:
        print(f" {var.name} = {var.x}")
```

```
Minimizing an LP problem

The original problem has:
    14 rows, 25 columns and 50 non-zero elements
The presolved problem has:
    14 rows, 25 columns and 50 non-zero elements

Starting the simplex solver using up to 8 threads

Problem info:
Range of matrix coefficients:    [1e+00,1e+00]
Range of rhs coefficients:       [0e+00,0e+00]
Range of bound coefficients:     [2e+01,5e+02]
Range of cost coefficients:      [1e+00,2e+01]

Method    Iteration           Objective  Primal.NInf   Dual.NInf        Time
Dual              0   -1.1765710549e+04           10           0       0.00s
Dual             12   -1.4596751067e+03            0           0       0.00s

Solving finished
Status: Optimal  Objective: -1.4600000000e+03  Iterations: 12  Time: 0.00s
Objective value: -1460.000000
Variable solution:
 s_m1 = 180.0
 s_m2 = 200.0
 s_m3 = 80.0
 m1_a1 = 50.0
 m1_b1 = 100.0
 m1_c1 = 0.0
 m1_m2 = 30.0
 m2_a2 = 75.0
 m2_b2 = 80.0
 m2_c2 = 75.0
 m2_m3 = 0.0
 m3_a3 = 0.0
 m3_b3 = 80.0
 m3_c3 = 0.0
 a1_t = 50.0
 b1_t = 100.0
 c1_t = 0.0
 a2_t = 75.0
 b2_t = 80.0
 c2_t = 75.0
 a3_t = 0.0
 b3_t = 80.0
 c3_t = 0.0
 s_t = 0.0
 bs = 460.0
```

report:

The maximum profit is $1460. The plan is to produce 180 units of plants during period 1 and sell 50 units at outlet A, sell 100 units at outlet B, and store 30 units; Produce 200 units of plants during period 2 and sell 75 units at outlet A, 80 units at outlet B, and 75 units at outlet C; Produce 80 units of plants during period 3 and sell 80 units at outlet B.

Question 5

objective function:

$$\min(\Sigma_{t=0}^{23} c_t x_t)$$

constraints:

$$x_t = \Sigma_t^{t+5} y_{t,(i\ mod\ 24)}, \quad t = 0, 1, \cdots, 23$$
$$\Sigma_{k=t-8}^{t} x_{k\ mod\ 24} - \Sigma_{k=t-5}^{t-3} y_{k\ mod\ 24,t} \geq r_t, \quad t = 0, 1, \cdots, 23$$
$$x_t \geq 0, \quad t = 0, 1, \cdots, 23$$