ISE 4623/5023: Deterministic Systems Models / Systems Optimization

University of Oklahoma
School of Industrial and Systems Engineering
Fall 2021

Individual assignment 8 (100 points)

NOTE: For all problems, you need to upload a PDF file of the solution, along with support files of any software used (Excel, Gurobi/Python, etc).

PLEDGE:

"On my honor, I affirm that I have neither given nor received inappropriate aid in the completion of this exercise."

Name: _____ ***Signature:*** _____

Student ID: _____ ***Date:*** _____

Problem TSP (50 points)

1. After 7 months of travel from Earth, on February 18, 2021, NASA's rover *Perseverance* landed on Mars, in the Jezero crater located in the Syrtis Major square of the martian surface. The rover is part of the Mars 2020 mission, and its purpose is to collect information on the habitability of the planet, search for possible microbial life, collect and store rock samples, among others. In the rock sample collection procedure, the mission control team divided the Syrtis Major into 15 quadrants in which the Perseverance will move to perform the respective task. The map of the region, as well as the quadrant labeling are shown in Figure 1:

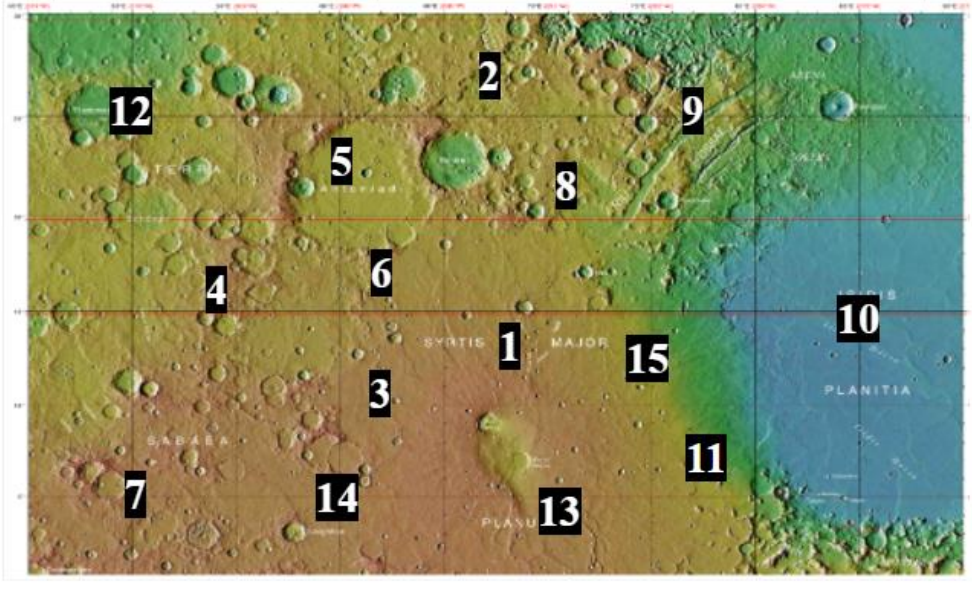


Figure 1.Figure 1: Quadrant of the Gulf of Sirte (Syrtis Major) of the Martian surface

Every two weeks (15 days), the Perseverance rover must collect samples in each quadrant. The recollection process in each quadrant lasts for one (1) day, time after which it will move to visit another quadrant (traveling time despicable). The martian's landscape poses a challenge for the rover to move between quadrants, but it will always be able to get the job done within a few hours. However, the amount of energy required to move between each pair of quadrants vary and is shown in Table 1:

Energy expenditure ($1 \cdot 10^6$ Joules)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
1	0	15.8	3.81	7.54	22.9	6.89	15.2	10.4	17.7	9.51	10.3	22.1	9.33	19.6	7.4
2	15.8	0	22.6	21.4	3.75	3.57	23.4	15.2	4.41	7.85	8.67	21.8	12.4	5.09	13.9
3	3.81	22.6	0	10	17.1	2.95	24	13.3	8.88	17.6	7.99	22.8	18.4	17.3	24
4	7.54	21.4	10	0	7.59	3.99	3.87	6.29	6.41	15.2	25.2	3.33	13.4	6.07	7.75
5	22.9	3.75	17.1	7.59	0	7.08	7.6	21.6	13.8	8.79	13	4.4	4.52	8.55	13.7
6	6.89	3.57	2.95	3.99	7.08	0	11.1	4.81	4.87	18.8	22.2	5.92	7.86	17.4	7.1
7	15.2	23.4	24	3.87	7.6	11.1	0	16.2	15.6	25.9	20.2	25.9	4.18	7.82	8.9
8	10.4	15.2	13.3	6.29	21.6	4.81	16.2	0	15.9	19.3	10.4	13.3	4.72	16.9	5.66
9	17.7	4.41	8.88	6.41	13.8	4.87	15.6	15.9	0	6.1	11	9.11	2.3	24.1	15.5
10	9.51	7.85	17.6	15.2	8.79	18.8	25.9	19.3	6.1	0	17.1	21.5	25.5	23.2	6.22
11	10.3	8.67	7.99	25.2	13	22.2	20.2	10.4	11	17.1	0	14	6.26	22.6	13.6
12	22.1	21.8	22.8	3.33	4.4	5.92	25.9	13.3	9.11	21.5	14	0	17.3	25.3	5.56
13	9.33	12.4	18.4	13.4	4.52	7.86	4.18	4.72	2.3	25.5	6.26	17.3	0	4.02	6.51
14	19.6	5.09	17.3	6.07	8.55	17.4	7.82	16.9	24.1	23.2	22.6	25.3	4.02	0	4.67
15	7.4	13.9	24	7.75	13.7	7.1	8.9	5.66	15.5	6.22	13.6	5.56	6.51	4.67	0

Table 1. Energy expenditure between each pair of quadrants

Last, no quadrants can be visited twice and all quadrants must be visited at least once during the two-week period. After the last quadrant has been visited and its associated one-day recollection process ends, the rover will go back to its starting quadrant to follow the same route, repeating the process bi-weekly during the mission's lifespan. Assume that any number of samples recollected are dully processed at the end of the 15^{th} day with a despicable amount of time and energy consumptions.

a) (25 points) Find the route that minimizes the bi-weekly total travel energy associated with visiting all 15 quadrants (and coming back to the initial quadrant). Assume that the Perseverance rover landed in quadrant "1" (so it would have to start and end its route there). Given this, note that the rover could either start the recollection in quadrant "1" in day 1, or could move and start the recollection in a different quadrant in day 1 (and start the recollection in quadrant "1" in its way back, in day 14). Mathematically formulate this problem (indicating sets, parameters, decision variables, objective function, and constraints. Include this formulation in your final report) and solve it using Gurobi (provide a proper snapshot of your code and optimization status).

- Sets:

N: Set of nodes $\{1,2,3,\dots,n\}$

A: Set of arcs

- Parameters:

c_{ij}: cost of traveling through arc $(i.j) \in A$

- Variables:

x_{ij}: binary variable that is 1 if salesman travels through arc $(i.j) \in A$, and is 0 otherwise.

u_i: label / order of visit of node $i \in N$ (to eliminate subtours)

- Objective function:

(o1) $$\min z = \sum_{(i,j)\in A} c_{ij}x_{ij}$$

- Constraints:

(c1) $$\sum_{i:(i,j)\in A} x_{ij} = 1 \qquad \forall j \in N$$

(c2) $$\sum_{j:(i,j)\in A} x_{ij} = 1 \qquad \forall i \in N$$

(c3) $$u_i \le u_j - 1 + n(1 - x_{ij}) \qquad \forall (i,j) \in A: j \neq 1$$

(c4) $$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A$$

(c5) $$u_i \ge 0 \qquad \forall i \in N$$

]:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	15.78	3.81	7.54	22.94	6.89	15.15	10.44	17.7	9.51	10.26	22.1	9.33	19.61	7.4
2	15.78	0	22.56	21.35	3.75	3.57	23.42	15.21	4.41	7.85	8.67	21.83	12.35	5.09	13.91
3	3.81	22.56	0	10.01	17.07	2.95	23.98	13.32	8.88	17.63	7.99	22.76	18.42	17.33	23.95
4	7.54	21.35	10.01	0	7.59	3.99	3.87	6.29	6.41	15.17	25.22	3.33	13.36	6.07	7.75
5	22.94	3.75	17.07	7.59	0	7.08	7.6	21.57	13.76	8.79	13.03	4.4	4.52	8.55	13.66
6	6.89	3.57	2.95	3.99	7.08	0	11.11	4.81	4.87	18.75	22.19	5.92	7.86	17.35	7.1
7	15.15	23.42	23.98	3.87	7.6	11.11	0	16.24	15.64	25.9	20.18	25.87	4.18	7.82	8.9
8	10.44	15.21	13.32	6.29	21.57	4.81	16.24	0	15.86	19.25	10.41	13.28	4.72	16.91	5.66
9	17.7	4.41	8.88	6.41	13.76	4.87	15.64	15.86	0	6.1	10.95	9.11	2.3	24.14	15.52
10	9.51	7.85	17.63	15.17	8.79	18.75	25.9	19.25	6.1	0	17.11	21.5	25.54	23.23	6.22
11	10.26	8.67	7.99	25.22	13.03	22.19	20.18	10.41	10.95	17.11	0	13.95	6.26	22.6	13.57
12	22.1	21.83	22.76	3.33	4.4	5.92	25.87	13.28	9.11	21.5	13.95	0	17.29	25.25	5.56
13	9.33	12.35	18.42	13.36	4.52	7.86	4.18	4.72	2.3	25.54	6.26	17.29	0	4.02	6.51
14	19.61	5.09	17.33	6.07	8.55	17.35	7.82	16.91	24.14	23.23	22.6	25.25	4.02	0	4.67
15	7.4	13.91	23.95	7.75	13.66	7.1	8.9	5.66	15.52	6.22	13.57	5.56	6.51	4.67	0

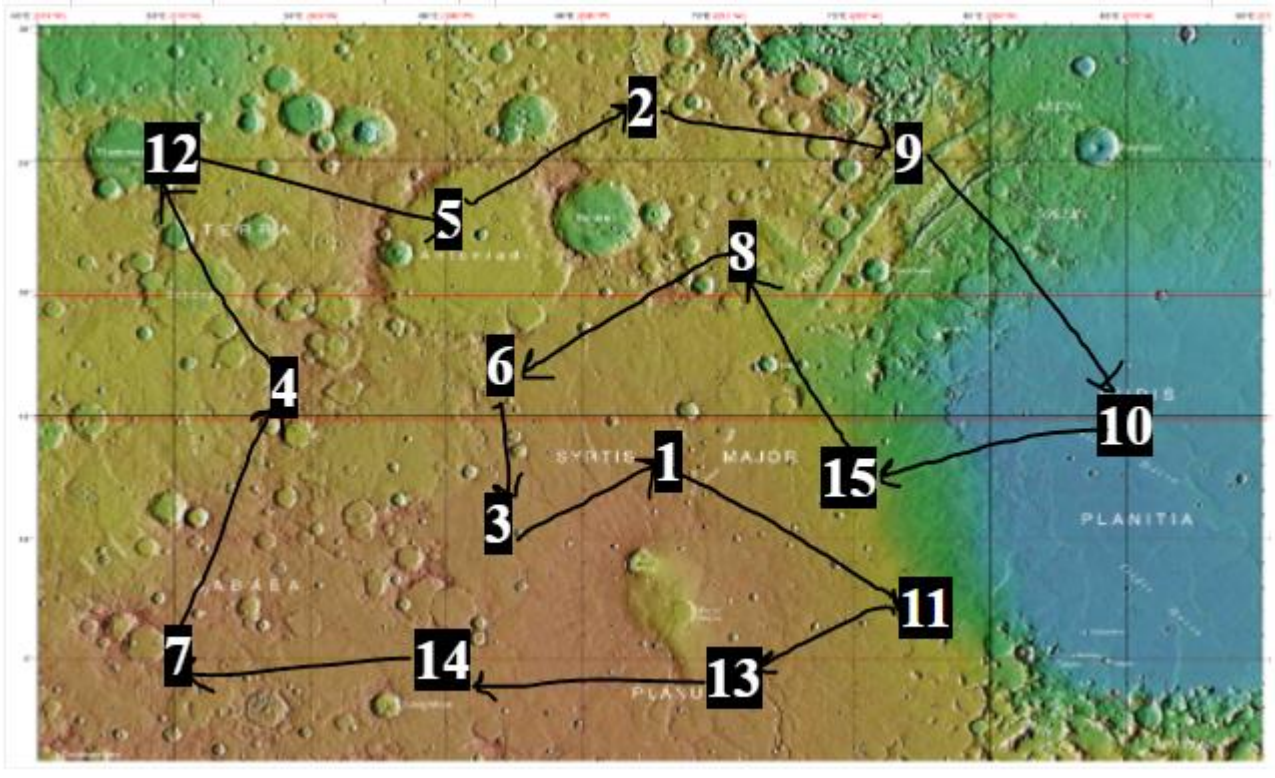
]:
```
A = [(i,j) for (i,j) in c.keys()]
M = 1e3
```

```
model_P3=Model('TSP')
model_P3.setParam(GRB.Param.OutputFlag, 0)
#Variables
x=model_P3.addVars(A,obj=c,vtype=GRB.BINARY,name="x")
u=model_P3.addVars(N,obj=0,name="u")

#Constraints
model_P3.addConstrs((x.sum('*',j) == 1 for j in N), "c1")
model_P3.addConstrs((x.sum(i,'*') == 1 for i in N), "c2")
model_P3.addConstrs((u[i] <= u[j]-1+n*(1-x[i,j]) for i,j in A if j!=1)

#Objective function
#model_P3.setParam('OutputFlag',1)
model_P3.optimize()

# Print solution
if model_P3.status == GRB.Status.OPTIMAL:
# #Basic printing
#     print('Obj: %g' % model_P3.objVal)
#     for v in model_P3.getVars():
#         print('%s %g' % (v.varName, abs(v.x)))

# #Nice printing
    print('')
    print('Obj: %g' % model_P3.objVal)
    print('')
    solution_x = model_P3.getAttr('x', x)
    solution_u = model_P3.getAttr('x', u)
    print('node: time of visit')
    for i in N:
        print('%s: %s' % (i, int(solution u[i])))
```

```
finished=0
start=1
end=1
print('')
print('start ---> end')
while finished==0:
    end=int(sum((j*solution_x[start,j] for j in N if (start,j) in A)))
    print('%s ---> %s' % (start, end))
    start=end
    if end==1:
        finished=1
```

```
Obj: 77.67

node: time of visit
1: 0
2: 9
3: 15
4: 6
5: 8
6: 14
7: 5
8: 13
9: 10
10: 11
11: 1
12: 7
13: 3
14: 4
15: 12

start ---> end
1 ---> 11
11 ---> 13
13 ---> 14
14 ---> 7
7 ---> 4
4 ---> 12
12 ---> 5
5 ---> 2
2 ---> 9
9 ---> 10
10 ---> 15
15 ---> 8
8 ---> 6
6 ---> 3
3 ---> 1
```

b) (12 points) Show your results, clearly indicating the obtained route (make your plot on top of Figure 1).



c) (13 points) Show the day in which the rover visits each quadrant and the total energy expenditure in millions of Joules (use Table 3 to report this) and discuss.

Monthly energy spent	$77.67 \cdot 10^6$ Joules
Quadrant	Day of visit
1	0
2	9
3	15
4	6
5	8
6	14
7	5
8	13
9	10
10	11
11	1
12	7
13	3
14	4
15	12

Table 2. Visit order format presentation

Problem 2 (40 points): (50 points + 20 extra credit points): Water distribution network design

It is the year 2050, and you are working for the water department for the State of New New York. The State of New New York has four megacities, that are supplied by three water sources and five treatment plants. Each

water source has a maximum production daily capacity (in millions of gallons) as shown in Figure 2. Similarly, each city has an associated water demand, as described in Figure 2. Each pipe (which connects either a water source with a treatment plant or a water plant with a city) has an associated daily flow cost (in millions of dollars per million gallons), a minimum daily flow to guarantee proper pressurization (in millions of gallons), and a maximum daily flow capacity (in millions of gallons), as shown in Figure 2 (these three values are shown in the same order).

a) (10 points) Develop an LP model to determine the minimum cost of satisfying the water demands of all the cities in the State of New York (indicating sets, parameters, decision variables, objective function, and constraints).

b_i: demand/supply for each node ($i \in N$)
c_{ij}: transportation cost associated with arc $(i, j) \in A$
u_{ij}: Maximum flow capacity associated with arc $(i, j) \in A$
l_{ij}: Minimum flow capacity, associated with arc $(i, j) \in A$

x_{ij}: the number of units of product through arc $(i, j) \in A$
d_i: unmet demand in node $i \in$ N
s_i: unmet demand in node $i \in$ N
λ_{ij}:Binary variable for if we are going to use link $(i, j) \in A$

$$Min \sum_{(i,j)\in A} c_{ij}x_{ij} + M\sum_{i\in N}(d_i+s_i)$$

$$\sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ij} = b_i + d_i - s_i \qquad \forall i \in N$$

$$x_{ij} \le u_{ij}\,\lambda_{ij} \qquad \forall (i,j) \in A$$

$$x_{ij} \ge l_{ij}\,\lambda_{ij} \qquad \forall (i,j) \in A$$

b) (10 points) Solve the model from part a), show your results, and discuss them in detail.

```
from gurobipy import *

# N -> set of nodes
# b -> demand/supply of each node
N, b = multidict({
    ('nodeC1'): -15,
    ('nodeC2'): -50,
    ('nodeC3'): -10,
    ('nodeC4'): -50,
    ('nodeT1'):0,
    ('nodeT2'):0,
    ('nodeT3'):0,
    ('nodeT4'):0,
    ('nodeT5'):0,
    ('nodeS1'): 75,
    ('nodeS2'): 150,
    ('nodeS3'): 50})
```

```
# A -> set of arcs:
# L -> flow lower bound for each arc
# U -> flow upper bound for each arc
# C -> flow unit cost for each arc


A, c, l, u = multidict({
    ('nodeS1', 'nodeT2'): [3, 1, 20],
    ('nodeS1', 'nodeT3'): [1, 3, 20],
    ('nodeS1', 'nodeT4'): [4, 1, 10],
    ('nodeS1', 'nodeT5'): [4, 1, 20],
    ('nodeS2', 'nodeT1'): [7, 15, 50],
    ('nodeS2', 'nodeT2'): [5, 5, 20],
    ('nodeS2', 'nodeT5'): [1, 2, 10],
    ('nodeS2', 'nodeT1'): [7, 15, 50],
    ('nodeS3', 'nodeT4'): [2, 5, 10],
    ('nodeS3', 'nodeT5'): [8, 2, 20],
    ('nodeT1', 'nodeC1'): [9, 10, 75],
    ('nodeT1', 'nodeC3'): [1, 5, 20],
    ('nodeT2', 'nodeC1'): [5, 3, 25],
    ('nodeT2', 'nodeC2'): [8, 1, 50],
    ('nodeT3', 'nodeC2'): [9, 5, 20],
    ('nodeT4', 'nodeC4'): [1, 1, 15],
    ('nodeT5', 'nodeC4'): [8, 3, 40]})

m = Model()

# var. for flow
x = m.addVars(A, name="x")

# var. for unmet demand and excessive supply
ud = m.addVars(N, name="ud")
es = m.addVars(N, name="es")

# Binary variable for if we are using the link or not
y = m.addVars(A, name="y", vtype=GRB.BINARY)

# Objective function
obj = sum(c[i, j]*x[i, j] for (i, j) in A) + 10000 * sum(es[i] + ud[i] for i in N)

# Balance constraint
m.addConstrs(
    (x.sum(i, '*')-x.sum('*', i) == b[i] - es[i] + ud[i] for i in N), "balanceConstraint")

# Arc upper bound constraint
m.addConstrs(
    (x[i, j] <= u[i, j]*y[i, j] for (i, j) in A), "UpperBoundConstraint")

# Arc lower bound constraint
m.addConstrs(
    (x[i, j] >= l[i, j]*y[i, j] for (i, j) in A), "LowerBoundConstraint")


m.setObjective(obj, GRB.MINIMIZE)
m.optimize()
m.printAttr('X')
```

```
Optimal solution found (tolerance 1.00e-04)
Best objective 1.501305000000e+06, best bound 1.501305000000e+06, gap 0.0000%

    Variable            X
-------------------------
x[nodeS1,nodeT2]           20
x[nodeS1,nodeT3]           20
x[nodeS1,nodeT4]            5
x[nodeS1,nodeT5]           20
x[nodeS2,nodeT1]           20
x[nodeS2,nodeT2]           15
x[nodeS2,nodeT5]           10
x[nodeS3,nodeT4]           10
x[nodeS3,nodeT5]            5
x[nodeT1,nodeC1]           10
x[nodeT1,nodeC3]           10
x[nodeT2,nodeC1]            5
x[nodeT2,nodeC2]           30
x[nodeT3,nodeC2]           20
x[nodeT4,nodeC4]           15
x[nodeT5,nodeC4]           35
  es[nodeS1]           10
  es[nodeS2]          105
  es[nodeS3]           35
y[nodeS1,nodeT2]            1
y[nodeS1,nodeT3]            1
y[nodeS1,nodeT4]            1
y[nodeS1,nodeT5]            1
y[nodeS2,nodeT1]            1
y[nodeS2,nodeT2]            1
y[nodeS2,nodeT5]            1
y[nodeS3,nodeT4]            1
y[nodeS3,nodeT5]            1
y[nodeT1,nodeC1]            1
y[nodeT1,nodeC3]            1
y[nodeT2,nodeC1]            1
y[nodeT2,nodeC2]            1
y[nodeT3,nodeC2]            1
y[nodeT4,nodeC4]            1
y[nodeT5,nodeC4]            1
```

c) (15 points) Develop an LP model to determine the maximum water demand satisfaction of all the cities in the State of New New York that can be achieved as a function of a number k, which indicated the maximum number of pipes that can be constructed/used. Hint: maximizing water demand satisfaction would be equivalent to minimizing total unsatisfied demand.

$$\sum_{j:(i,j)\in A} \lambda_{ij} \leq k \qquad \forall (i,j) \in A$$

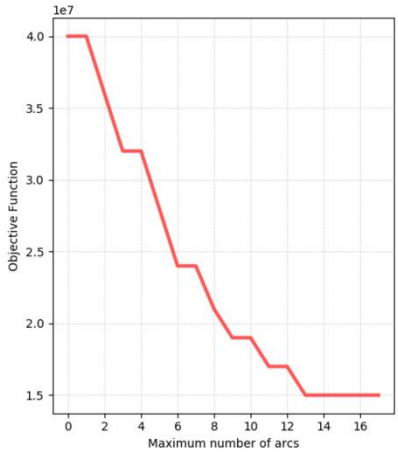
d) (15 points) Solve the model from part c), assuming k=7. Indicate the maximum total demand satisfaction that can be achieved, and the cost associated with the obtained solution. Show your results, and discuss them in detail.

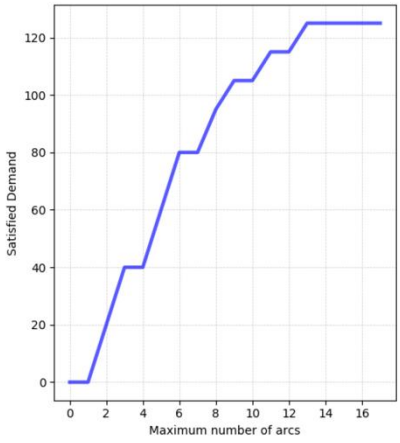
```
k = 7


m = Model()


# var. for flow

x = m.addVars(A, name="x")


# var. for unmet demand and excessive supply

ud = m.addVars(N, name="ud")

es = m.addVars(N, name="es")


# Binary variable for if we are using the link or not

y = m.addVars(A, name="y", vtype=GRB.BINARY)
```

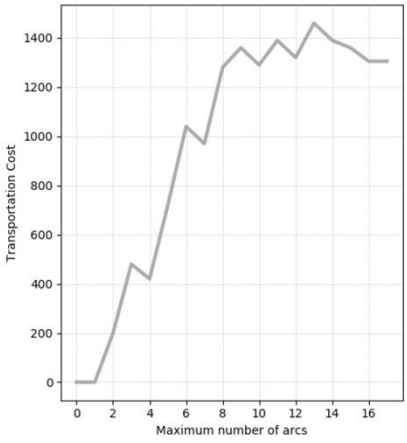
```
# Objective function
obj = sum(c[i, j]*x[i, j] for (i, j) in A) + 10000 * sum(es[i] + ud[i] for i in N)


# Balance constraint
m.addConstrs(
    (x.sum(i, '*')-x.sum('*', i) == b[i] - es[i] + ud[i] for i in N), "balanceConstraint")


# Arc upper bound constraint
m.addConstrs(
    (x[i, j] <= u[i, j]*y[i, j] for (i, j) in A), "UpperBoundConstraint")


# Arc lower bound constraint
m.addConstrs(
    (x[i, j] >= l[i, j]*y[i, j] for (i, j) in A), "LowerBoundConstraint")


# Upper bound for the number of arcs that can be used
m.addConstr(sum(y[i, j] for (i, j) in A) <= k, "max arc number")


m.setObjective(obj, GRB.MINIMIZE)
m.optimize()
m.printAttr('X')
```

```
Optimal solution found (tolerance 1.00e-04)
Best objective 2.400980000000e+06, best bound 2.400801500000e+06, gap 0.0074%

    Variable            X
-------------------------
x[nodeS1,nodeT2]           20
x[nodeS1,nodeT3]           20
x[nodeS1,nodeT5]           20
x[nodeS3,nodeT5]           20
x[nodeT2,nodeC2]           20
x[nodeT3,nodeC2]           20
x[nodeT5,nodeC4]           40
  ud[nodeC1]           15
  ud[nodeC2]           10
  ud[nodeC3]           10
  ud[nodeC4]           10
  es[nodeS1]           15
  es[nodeS2]          150
  es[nodeS3]           30
y[nodeS1,nodeT2]            1
y[nodeS1,nodeT3]            1
y[nodeS1,nodeT5]            1
y[nodeS3,nodeT5]            1
y[nodeT2,nodeC2]            1
y[nodeT3,nodeC2]            1
y[nodeT5,nodeC4]            1

Process finished with exit code 0
```

e) (20 points Extra credit) Construct a plot of the maximum total water demand satisfaction that can be achieved as a function of k, for k = {0, 1, 2, ..., 17}, and a separate plot of the total flow costs associated with each maximum demand satisfaction (also as a function of k). Discuss your results in detail.


1e7
Objective Function
Maximum number of arcs


Satisfied Demand
Maximum number of arcs


Transportation Cost
Maximum number of arcs