MSOR yiping.t@columbia.edu

Yi Ping Tseng yt2690.

prob 1.

1. $\min\ 40y_1 + 15y_2 + 30y_3$

$2y_1 + y_2 + y_3 \geq 2$

$y_1 - y_2 + y_3 \leq -1$

$y_1 + 2y_2 - y_3 = 1$

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

2. ⓐ $(25, 0, -5)^T$:

$\Rightarrow$ Since $x_2 = 0$ $\Rightarrow$ leave us equation 1 & 3 in Dual

$\Rightarrow$ Since $\begin{cases} 2\times 25 + 0 + (-5) \geq 40 & \rightarrow \text{not tight} \\ 1\times 25 - 0 + 2(-5) = 15 & \rightarrow \text{tight} \\ 1\times 25 + 0 - (-5) \leq 30 & \rightarrow \text{tight} \end{cases}$

$\Rightarrow \begin{cases} 2y_1 + y_2 + y_3 = 2 \\ y_1 + 2y_2 - y_3 = 1 \\ y_1 = 0 \end{cases}$

$\Rightarrow \begin{cases} y_2 + y_3 = 2 \\ 2y_2 - y_3 = 1 \end{cases}$

$\Rightarrow$ $y_1 = 0$ $y_2 = 1$ $y_3 = 1$ $\Rightarrow$ it's infeasible in $y_1 - y_2 + y_3 \leq -1$ #

ⓑ $(40, -45, -35)^T$

$\Rightarrow$ Since $\begin{cases} 2\times 40 + (-45) + (-35) = 40 \geq 40 & \rightarrow \text{tight} \\ 1\times 40 - (-45) + 2\cdot(-35) = 15 = 15 & \rightarrow \text{tight} \\ 1\times 40 + (-45) - (-35) = 30 \leq 30 & \rightarrow \text{tight} \end{cases}$

$\Rightarrow \begin{cases} 2y_1 + y_2 + y_3 = 2 \\ y_1 - y_2 + y_3 = -1 \\ y_1 + 2y_2 - y_3 = 1 \end{cases}$

$\Rightarrow$ $y_1 = -\frac{3}{2}$ $y_2 = -3$ $y_3 = \frac{9}{2}$ ⊛

ⓒ. $(13, 0, 1)^T$:

$$\Rightarrow \begin{cases} 3 \cdot 13 + 0 + 1 \cdot 1 = 40 \geq 40 & \longrightarrow \text{tight} \\ 1 \cdot 13 - 0 + 2 \cdot 1 = 15 = 15 & \longrightarrow \text{tight} \\ 1 \cdot 13 + 0 - 1 \cdot 1 = 12 \leq 30 & \longrightarrow \text{not tight} \end{cases}$$

$$\Rightarrow \begin{cases} 3y_1 + y_2 + y_3 = 2 \\ y_1 + 2y_2 - y_3 = 1 \\ y_3 = 0 \end{cases}$$

$$6y_1 + 2y_2 = 4$$
$$y_1 + 2y_2 = 1$$

$$\Rightarrow y_1 = \frac{3}{5} \Rightarrow \text{invalid} \quad y_1 \leq 0$$

there is no optimal solution for the dual, given this primal solution. #

prob 2

(a) Dual

$\min\ 8y_1 + 10y_2$

$2y_1 + 3y_2 \geq 3.$

$5y_1 + 7y_2 \geq 2$

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

given the primal bfs $(\frac{10}{3}, 0)$

$\Rightarrow \begin{cases} 2\cdot\frac{10}{3} + 0 = \frac{20}{3} \leq 8 \longrightarrow \text{not tight} \\ 3\cdot\frac{10}{3} + 0 = \frac{30}{3} = 10 \leq 10. \longrightarrow \text{tight} \end{cases}$

$\Rightarrow \begin{cases} 2y_1 + 3y_2 = 3 \\ y_1 = 0 \end{cases}$

$\Rightarrow y_1 = 0 \quad y_2 = 1$

(b) $s_1 = 8 - 2x_1 - 5x_2$

$s_2 = 10 - 3x_1 - 7x_2$

$z = 3x_1 + 2x_2$

$\Rightarrow \begin{cases} s_1 = 8 - 2x_1 - 5x_2 \\ s_2 - \Delta = 10 - 3x_1 - 7x_2 \\ z = 3x_1 + 2x_2 \end{cases}$

$\Rightarrow$ by tableau

$\begin{cases} z = 10 - 5x_2 - s_2 \\ x_1 = \frac{10}{3} - \frac{7}{3}x_2 - \frac{1}{3}s_2 \\ s_1 = \frac{4}{3} - \frac{1}{3}x_2 + \frac{2}{3}s_2 \end{cases} \Rightarrow \begin{cases} z = (10 + \Delta) - 5x_2 - s_2 \\ x_1 = (\frac{10}{3} + \frac{\Delta}{3}) - \frac{7}{3}x_2 - \frac{1}{3}s_2 \\ s_1 = (\frac{4}{3} - \frac{2}{3}\Delta) - \frac{1}{3}x_2 + \frac{2}{3}s_2 \end{cases}$

$\begin{cases} \frac{10}{3} + \frac{\Delta}{3} \geq 0 \\ \frac{4}{3} - \frac{2}{3}\Delta \geq 0 \end{cases} \Rightarrow \begin{cases} \Delta \geq -10. \\ 2 \geq \Delta \end{cases} \Rightarrow 3x_1 + 7x_2 \leq 10 + \Delta$

$\rightarrow$ so the range of $b_2 = 10 - \Delta \Rightarrow 12 \geq b \geq 0.$ *

Prob 2.

(b) if $b_2' = 5$

$\rightarrow b_2' = 5 = 10 + \Delta \Rightarrow \Delta = -5$

$x_1 = \frac{10}{3} + \frac{-5}{3} = \frac{5}{3}$ $\qquad x_2 = s_2 = 0.$

$s_1 = \frac{4}{3} - \frac{2}{3} \cdot (-5) = \frac{14}{3}.$

$z = 3x_1 + 2x_2 = 3 \cdot \frac{5}{3} + 0 = 5$ #

c. Since the constraints of Dual y_i are $y_i \geq 0$.

and since $x_i \geq 0$ ⇒ all equations in Dual should larger than the coefficient of Primal objective

if we change the coefficient of Primal objective to

< 0 ⇒ then all Dual equation are larger than a negative value

> 0 ⇒ this is the same question of original problem, there might be an y_i that satisified the constraints and equations.

⇒ we cannot change the obj function of Primal to make Dual infeasible

prob 2.

ⓐ Dual

$$\min \; 2y_1 + 8y_2$$

$$8y_1 + 6y_2 \geq 4$$

$$2y_1 + y_2 \geq 1.$$

$$y_1 + y_2 \geq 2$$

given bfs $= (0, 0, 2)$ in Primal

$$\Rightarrow \begin{cases} 0 + 0 + 2 = 2 \leq 2 & \rightarrow \text{tight} \\ 0 + 0 + 2 = 2 \leq 8 & \rightarrow \text{not tight} \end{cases}$$

$$\Rightarrow \begin{cases} y_1 + y_2 = 2 \\ y_2 = 0 \end{cases}$$

$$\Rightarrow \quad y_1 = 2 \quad y_2 = 0 \quad \#$$

Prob 3.

(b) $z = 4x_1 + x_2 + c_3 x_3$

$x_3 = 2 - 8x_1 - 3x_2 - S_1$

$S_2 = 6 + 2x_1 + 2x_2 + S_1$

$\Rightarrow z' = 4x_1 + x_2 + c_3(2 - 8x_1 - 3x_2 - S_1)$

$= 2c_3 + (4 - 8c_3)x_1 + (1 - 3c_3)x_2 - c_3 S_1$

$$\begin{cases} 4 - 8c_3 < 0 \\ 1 - 3c_3 < 0 \\ -c_3 < 0 \end{cases} \Rightarrow \begin{cases} c_3 > \frac{1}{2} \\ c_3 > \frac{1}{3} \\ c_3 > 0. \end{cases}$$

$\Rightarrow$ the range of x_3 coefficient for which the current basis remains optimal is $c_3 \geq \frac{1}{2}$ #

(c) $z = c_1 x_1 + x_2 + 2x_3$.

$x_3 = 2 - 8x_1 - 3x_2 - S_1$

$S_2 = 6 + 2x_1 + 2x_2 + S_1$

$\Rightarrow z' = c_1 x_1 + x_2 + 2(2 - 8x_1 - 3x_2 - S_1)$

$= 4 + (c_1 - 16)x_1 - 2x_2 - 2S_1$

$\Rightarrow c_1 - 16 < 0 \Rightarrow c_1 \leq 16$ #

Prob 4 ⓐ

$$A = \begin{array}{c|ccc} & P & S & R \\ \hline P & 0 & 1 & -1 \\ S & -1 & 0 & 1 \\ R & 1 & -1 & 0 \end{array}$$

for row player optimal strategy = (x_1, x_2, x_3).

$$\begin{aligned} \max \quad & V \quad \text{(maximize minimum expected profit)} \\ \text{s.t.} \quad & 0 - x_2 + x_3 - V \geq 0. \\ & x_1 + 0 - x_3 - V \geq 0. \\ & -x_1 + x_2 + 0 - V \geq 0 \\ & x_1 + x_2 + x_3 = 1 \\ & x_1, x_2, x_3, V \geq 0 \end{aligned}$$

for column player optimal strategy = (y_1, y_2, y_3)

$$\begin{aligned} \min \quad & u \quad \text{(minimize maximum expected loss)} \\ \text{s.t.} \quad & 0 + y_2 - y_3 - u \leq 0. \\ & -y_1 + 0 + y_3 - u \leq 0 \\ & y_1 - y_2 + 0 - u \leq 0. \\ & y_1 + y_2 + y_3 = 1 \\ & y_1, y_2, y_3, u \geq 0. \end{aligned}$$

by gaurobi : $(x_1, x_2, x_3, V) = (0.\bar{3}, 0.\bar{3}, 0.\bar{3}, 0)$

$(y_1, y_2, y_3, u) = (0.\bar{3}, 0.\bar{3}, 0.\bar{3}, 0)$ #.

```python
from gurobipy import *


# create a model
m = Model("problem 4 for the row player")

# create variables
x1 = m.addVar(vtype=GRB.CONTINUOUS, name="x1", lb=0)
x2 = m.addVar(vtype=GRB.CONTINUOUS, name="x2", lb=0)
x3 = m.addVar(vtype=GRB.CONTINUOUS, name="x3", lb=0)
x4 = m.addVar(vtype=GRB.CONTINUOUS, name="x4", lb=0)


# integrate new variables
m.update()

# set objective
m.setObjective(
    x4,
    GRB.MAXIMIZE
)

# add constraints
m.addConstr(-1*x2 + x3 - x4 >= 0)
m.addConstr(x1 - x3 - x4 >= 0)
m.addConstr(-1*x1 + x2 - x4 >= 0)
m.addConstr(x1 + x2 + x3 == 1)

# optimize
m.optimize()
print("Model status: ", m.status)

# print out decision variables
for v in m.getVars():
    print(v.varName, v.x, "\n")

print("-"*15)
print("Obj Value: ", m.objVal)
```

```
tsengyiping@cengyipings-MacBook-Air: ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework...   ⌥⌘1
┌tsengyiping@cengyipings-MacBook-Air ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework3 <master*>
└$ python hw3-prob4-rp.py
Academic license - for non-commercial use only
Optimize a model with 4 rows, 4 columns and 12 nonzeros
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [1e+00, 1e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+00, 1e+00]
Presolve removed 1 rows and 1 columns
Presolve time: 0.00s
Presolved: 3 rows, 3 columns, 9 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    7.5125000e-01   1.626875e+00   0.000000e+00      0s
       2   -0.0000000e+00   0.000000e+00   0.000000e+00      0s

Solved in 2 iterations and 0.00 seconds
Optimal objective -0.000000000e+00
('Model status: ', 2)
('x1', 0.3333333333333333, '\n')
('x2', 0.3333333333333333, '\n')
('x3', 0.3333333333333333, '\n')
('x4', 0.0, '\n')
---------------
('Obj Value: ', -0.0)
┌tsengyiping@cengyipings-MacBook-Air ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework3 <master*> ┌t
┌tsengyiping@cengyipings-MacBook-Air ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework3 <master*>
└$
```

```
from gurobipy import *


# create a model
m = Model("problem 4 for the column player")

# create variables
y1 = m.addVar(vtype=GRB.CONTINUOUS, name="y1", lb=0)
y2 = m.addVar(vtype=GRB.CONTINUOUS, name="y2", lb=0)
y3 = m.addVar(vtype=GRB.CONTINUOUS, name="y3", lb=0)
y4 = m.addVar(vtype=GRB.CONTINUOUS, name="y4", lb=0)


# integrate new variables
m.update()

# set objective
m.setObjective(
    y4,
    GRB.MINIMIZE
)

# add constraints
m.addConstr(y2 - y3 - y4 <= 0)
m.addConstr(-1*y1 + y3 - y4 <= 0)
m.addConstr(y1 - y2 - y4 <= 0)
m.addConstr(y1 + y2 + y3 == 1)

# optimize
m.optimize()
print("Model status: ", m.status)

# print out decision variables
for v in m.getVars():
    print(v.varName, v.x, "\n")

print("-"*15)
print("Obj Value: ", m.objVal)
```

tsengyiping@cengyipings-MacBook-Air: ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework... ⌥⌘1

```
┌tsengyiping@cengyipings-MacBook-Air ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework3 ‹master*›
└$ python hw3-prob4-cp.py
Academic license - for non-commercial use only
Optimize a model with 4 rows, 4 columns and 12 nonzeros
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [1e+00, 1e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+00, 1e+00]
Presolve time: 0.00s
Presolved: 4 rows, 4 columns, 12 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    0.0000000e+00   1.000000e+00   0.000000e+00      0s
       2    0.0000000e+00   0.000000e+00   0.000000e+00      0s

Solved in 2 iterations and 0.00 seconds
Optimal objective  0.000000000e+00
('Model status: ', 2)
('y1', 0.3333333333333333, '\n')
('y2', 0.33333333333333337, '\n')
('y3', 0.33333333333333337, '\n')
('y4', 0.0, '\n')
---------------
('Obj Value: ', 0.0)
┌tsengyiping@cengyipings-MacBook-Air ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework3 ‹master*›
└$
```

prob 4 (b)

$$A = \begin{array}{c|ccc|c} & P & S & R & \\ \hline & 0 & 1 & -1 & P \\ & -1 & 0 & 1 & S \\ & 1 & -1 & 0 & R \end{array}$$

if RP $\bar{x} = \left(\frac{5}{10}, \frac{1}{10}, \frac{4}{10}\right)$

the expected loss for CP to play

Paper : $\frac{5}{10} \times 0 + \frac{1}{10} \times (-1) + \frac{4}{10} \times (1) = \frac{3}{10}$.

Scissor : $\frac{5}{10} \times 1 + \frac{1}{10} \times 0 + \frac{4}{10} \times (-1) = \frac{1}{10}$.

Rock : $\frac{5}{10} \times (-1) + \frac{1}{10} \times 1 + \frac{4}{10} \times 0 = \frac{-4}{10}$

if CP want to win this game.

he would like to min Less

So he would choose Rock all the time

and the expected loss is $-\frac{4}{10}$

$$\boxed{\text{CP} : \begin{array}{c} \ \ P \quad S \quad R \\ (0, \ 0, \ 1) \end{array} \qquad \text{expected loss} : -\frac{4}{10}.\ \text{※}}$$

$$\begin{aligned} \min \ & \bar{x} A y \\ & \sum y_i = 1 \\ & y_i \geq 0. \end{aligned} \Rightarrow \begin{aligned} \min \ & \frac{3}{10} y_1 + \frac{1}{10} y_2 - \frac{4}{10} y_3. \\ \text{s.t.} \ & y_1 + y_2 + y_3 = 1 \\ & y_1, y_2, y_3 \geq 0 \end{aligned}$$

$\Rightarrow$ opt value: $-\frac{4}{10}$

strategy: $\begin{array}{c} P \quad S \quad R \\ (0, \ 0, \ 1) \end{array}$

prob 5

$$A = \begin{pmatrix} 2 & -1 & 3 & -2 \\ 1 & 4 & -3 & 0 \\ 0 & -2 & -1 & 3 \end{pmatrix}$$

A_{ij} = row player gain A_{ij}

column player lose A_{ij}.

for Row Player $\rightarrow$ (x_1, x_2, x_3) $x_i :=$ the prob to play strategy i

P_i : the profit of row player if column player plays strategy i

the all possible expected profit for RP.

$P_1 = 2x_1 + x_2 + 0$

$P_2 = -x_1 + 4x_2 - 2x_3.$

$P_3 = 3x_1 - 3x_2 + x_3$

$P_4 = -2x_1 + 0 + 3x_3.$

we know the expected profit for RP is $\overset{\text{at least}}{\vee}$ $\min(P_1, P_2, P_3, P_4) = V$

obj : max V where $V = \min(P_1, P_2, P_3, P_4)$

$$\begin{aligned}
\text{s.t.} \quad & 2x_1 + x_2 + 0 - V \geq 0 \\
& -x_1 + 4x_2 - 2x_3 - V \geq 0 \\
& 3x_1 - 3x_2 - x_3 - V \geq 0 \\
& -2x_1 + 0 + 3x_3 - V \geq 0 \\
& x_1 + x_2 + x_3 = 0. \\
& x_1, x_2, x_3, V \geq 0.
\end{aligned}$$

for Column Player

(y_1, y_2, y_3, y_4) $y_i :=$ the prob to play strategy i

P_i : the expected loss ~~of~~ of column player if row player plays i

$P_1 = 2y_1 - y_2 + 3y_3 - 2y_4$

$P_2 = y_1 + 4y_2 - 3y_3 + 0 \cdot y_4$

$P_3 = 0 \cdot y_1 - 2y_2 - y_3 + 3y_4$

we know the expected loss for CP is $\overset{\text{at most}}{\downarrow}$ $\max(P_1, P_2, P_3)$. $= u$

obj : $\min \max(P_1, P_2, P_3)$.

$$\Rightarrow \min \; u$$

$$\begin{aligned}
\text{s.t.} \quad & 2y_1 - y_2 + 3y_3 - 2y_4 - u \le 0 \\
& y_1 + 4y_2 - 3y_3 + 0 - u \le 0 \\
& 0 - 2y_2 - y_3 + 3y_4 - u \le 0 \\
& y_1 + y_2 + y_3 + y_4 = 1 \\
& y_1, y_2, y_3, y_4, u \ge 0
\end{aligned}$$

by gaurobi,

RP : $(x_1, x_2, x_3, v) = (0.41\bar{3}, 0.28, 0.30\bar{6}, 0.09\bar{3})$.

CP : $(y_1, y_2, y_3, y_4, u) = (0, 0.29\bar{3}, 0.36, 0.34\bar{6}, 0.09\bar{3})$.

```
from gurobipy import *


# create a model
m = Model("problem 5 for the row player")

# create variables
x1 = m.addVar(vtype=GRB.CONTINUOUS, name="x1", lb=0)
x2 = m.addVar(vtype=GRB.CONTINUOUS, name="x2", lb=0)
x3 = m.addVar(vtype=GRB.CONTINUOUS, name="x3", lb=0)
x4 = m.addVar(vtype=GRB.CONTINUOUS, name="x4", lb=0)

# integrate new variables
m.update()

# set objective
m.setObjective(
    x4,
    GRB.MAXIMIZE
)

# add constraints
m.addConstr(2*x1 + 1*x2 + 0*x3 - x4 >= 0)
m.addConstr(-1*x1 + 4*x2 - 2*x3 - x4 >= 0)
m.addConstr(3*x1 - 3*x2 - 1*x3 - x4 >= 0)
m.addConstr(-2*x1 + 0*x2 + 3*x3 - x4 >= 0)
m.addConstr(x1 + x2 + x3 == 1)

# optimize
m.optimize()
print("Model status: ", m.status)

# print out decision variables
for v in m.getVars():
    print(v.varName, v.x, "\n")

print("-"*15)
print("Obj Value: ", m.objVal)
```

```
tsengyiping@cengyipings-MacBook-Air: ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework...   ⌥⌘1
┌─tsengyiping@cengyipings-MacBook-Air ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework3 <master*>
└─$ python hw3-prob5-rp.py
Academic license - for non-commercial use only
Optimize a model with 5 rows, 4 columns and 17 nonzeros
Coefficient statistics:
  Matrix range     [1e+00, 4e+00]
  Objective range  [1e+00, 1e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+00, 1e+00]
Presolve removed 1 rows and 1 columns
Presolve time: 0.00s
Presolved: 4 rows, 3 columns, 12 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    1.6178000e+00   1.179611e+00   0.000000e+00      0s
       3    9.3333333e-02   0.000000e+00   0.000000e+00      0s

Solved in 3 iterations and 0.00 seconds
Optimal objective  9.333333333e-02
('Model status: ', 2)
('x1', 0.41333333333333333, '\n')
('x2', 0.28, '\n')
('x3', 0.30666666666666664, '\n')
('x4', 0.09333333333333338, '\n')
---------------
('Obj Value: ', 0.09333333333333338)
┌─tsengyiping@cengyipings-MacBook-Air ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework3 <master*>
└─$
```

```
from gurobipy import *


# create a model
m = Model("problem 5 for the column player")

# create variables
y1 = m.addVar(vtype=GRB.CONTINUOUS, name="y1", lb=0)
y2 = m.addVar(vtype=GRB.CONTINUOUS, name="y2", lb=0)
y3 = m.addVar(vtype=GRB.CONTINUOUS, name="y3", lb=0)
y4 = m.addVar(vtype=GRB.CONTINUOUS, name="y4", lb=0)
y5 = m.addVar(vtype=GRB.CONTINUOUS, name="y5", lb=0)

# integrate new variables
m.update()

# set objective
m.setObjective(
    y5,
    GRB.MINIMIZE
)

# add constraints
m.addConstr(2*y1 - y2 + 3*y3 - 2*y4 - y5 <= 0)
m.addConstr(y1 + 4*y2 - 3*y3 + 0*y4 - y5 <= 0)
m.addConstr(0*y1 - 2*y2 - 1*y3 + 3*y4 - y5 <= 0)
m.addConstr(y1 + y2 + y3 + y4 == 1)

# optimize
m.optimize()
print("Model status: ", m.status)

# print out decision variables
for v in m.getVars():
    print(v.varName, v.x, "\n")

print("-"*15)
print("Obj Value: ", m.objVal)
```

```
tsengyiping@cengyipings-MacBook-Air: ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework...   ⌥⌘1
┌─tsengyiping@cengyipings-MacBook-Air ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework3 ‹master*›
└─$ python hw3-prob5-cp.py
Academic license - for non-commercial use only
Optimize a model with 4 rows, 5 columns and 17 nonzeros
Coefficient statistics:
  Matrix range     [1e+00, 4e+00]
  Objective range  [1e+00, 1e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+00, 1e+00]
Presolve time: 0.00s
Presolved: 4 rows, 5 columns, 17 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    0.0000000e+00   7.500000e-01   0.000000e+00      0s
       4    9.3333333e-02   0.000000e+00   0.000000e+00      0s

Solved in 4 iterations and 0.00 seconds
Optimal objective  9.333333333e-02
('Model status: ', 2)
('y1', 0.0, '\n')
('y2', 0.29333333333333333, '\n')
('y3', 0.36, '\n')
('y4', 0.3466666666666667, '\n')
('y5', 0.09333333333333333, '\n')
---------------
('Obj Value: ', 0.09333333333333333)
┌─tsengyiping@cengyipings-MacBook-Air ~/OneDrive/Columbia/MSOR/Optimization Models and Methods/Optimization-Homework/homework3 ‹master*›
└─$
```

prob 6.

1. we know if there exists an negative cycle, the label-correction algorithm will enter an infinite loop.

2. we also know if there are no negative cycles, the maximum of loop is $|V| - 1$ times.

3. if we want to know whether there is a negative cycle or not, we should count the # of times the label correcting algo scans its path weight.

if the # of scanning times $\leq |V| - 1$: no negative cycle

if the # of scaning times $\geq |V|$: there is at least one negative cycle.

prob 7.

① we know there is a weight w_{ij} between i currency and j currency, which means 1 dollar i can exchange w_{ij} dollar j.

② our obj function is to find an exchange path to end up with more than the original dollar

ex. $\$ \rightarrow A \rightarrow B \rightarrow C \rightarrow \$$

$100\, w_{\$A}\, w_{AB}\, w_{BC}\, w_{C\$} = 101\$$ ← our goal is maximize this value

③ after log each weight, it is a longest path problem

④ to find the longest path → we can multiple (-1) to each $\log(w_{ij})$ and turn the problem into the shortest path

⑤ we know there are negative weights and we are finding the shortest path

→ we can use label correcting algorithm to find the shortest path among all $-\log(w_{ij})$

⑥ In each iteration, we have to count the # of scans

Once the # of scans = the # of weights, $|W|$,

it means there is a negative cycles, we directly break the loop.

⑦ algo result

a. the sum of shortest path > 0 ⇒ no way to exchange more than 100\$

b. the sum of shortest path < 0 ⇒ it is a path to exchange over 100\$.

c. there is a negative cycle ⇒ we can get extreme wealth