R10521516 廖浚評 OR_HW8

```python
from gurobipy import *
def addConstr(n):
    if n ==1:
        return
    if n ==2:
        return m.addConstr( x2  >= 6, "c2")
    if n ==3:
        return m.addConstr( x2  >= 6, "c2") ,m.addConstr( x1  <= 1, "c3")
    if n ==4:
        return m.addConstr( x2  >= 6, "c2") ,m.addConstr( x1  <= 1, "c3"),m.addConstr( x2  <= 6, "c4")
    if n ==5:
        return m.addConstr( x2  >= 6, "c2") ,m.addConstr( x1  <= 1, "c3"),m.addConstr( x2  >= 7, "c4")
    if n ==6:
        return m.addConstr( x2  >= 6, "c2") ,m.addConstr( x1  >= 2, "c3")
    if n ==7:
        return m.addConstr( x2  <= 5, "c2")

for _ in range(7):
    print(_+1)
    m = Model("LP")
    x1 = m.addVar( name="x1")
    x2 = m.addVar( name="x2")
    m.setObjective(2*x1 + 3*x2 , GRB.MAXIMIZE)
    m.update()
    m.addConstr(2* x1 +  1* x2 <= 10, "c0")
    m.addConstr( 15 * x1 + 30 * x2 <= 200, "c1")
    m.addConstr(x1 >= 0)
    m.addConstr(x2 >= 0)
    m.update()
    addConstr(_+1)
    m.update()
    m.setParam( 'OutputFlag', False )
    m.optimize()
    if m.status == GRB.OPTIMAL:
        print('Optimal objective: %g' % m.objVal)
    elif m.status == GRB.INF_OR_UNBD:
        print('Model is infeasible or unbounded')
        continue
    elif m.status == GRB.INFEASIBLE:
        print('Model is infeasible')
        continue
    elif m.status == GRB.UNBOUNDED:
        print('Model is unbounded')
        continue
    else:
        print('Optimization ended with status %d' % m.status)
        continue
    for v in m.getVars():
        print('%s: %f' % (v.varName, v.x))
```

```
Console 1/A

In [267]: runfile('C:/Users/paddy/
homework8.py', wdir='C:/Users/paddy
1
Optimal objective: 21.1111
x1: 2.222222
x2: 5.555556
2
Optimal objective: 20.6667
x1: 1.333333
x2: 6.000000
3
Optimal objective: 20.5
x1: 1.000000
x2: 6.166667
4
Optimal objective: 20
x1: 1.000000
x2: 6.000000
5
Model is infeasible or unbounded
6
Model is infeasible or unbounded
7
Optimal objective: 20
x1: 2.500000
x2: 5.000000
```

沒看到要在同一個.py 實現所有的 subproblem，這是後來改的，跟下面那些之前寫的步驟一樣

```python
from gurobipy import *
m = Model("LP")
'''
x1 = m.addVar( name="x1", vtype=GRB.INTEGER)
x2 = m.addVar( name="x2", vtype=GRB.INTEGER)
'''
x1 = m.addVar( name="x1")
x2 = m.addVar( name="x2")
m.setObjective(2*x1 + 3*x2 , GRB.MAXIMIZE)
m.update()
m.addConstr(2* x1 +  1* x2 <= 10, "c0")
m.addConstr( 15 * x1 + 30 * x2 <= 200, "c1")
m.addConstr(x1 >= 0)
m.addConstr(x2 >= 0)
m.update()

#用來算枚舉數的限制式
'''
m.addConstr( x2  >= 6, "c2")
m.addConstr( x1  <= 1, "c3")
m.addConstr( x2  >= 6, "c4")
'''
m.setParam( 'OutputFlag', False )

m.optimize()

print('Optimal :')
for v in m.getVars():
    print('%s: %f' % (v.varName, v.x))
print('Obj: %f' % m.objVal)
```

```
Optimal :
x1: 2.222222
x2: 5.555556
Obj: 21.111111
```

一開始沒有設定限制

向下設定 X2>=6 X2<=5

```
from gurobipy import *
m = Model("LP")
'''
x1 = m.addVar( name="x1", vtype=GRB.INTEGER)
x2 = m.addVar( name="x2", vtype=GRB.INTEGER)
'''
x1 = m.addVar( name="x1")
x2 = m.addVar( name="x2")
m.setObjective(2*x1 + 3*x2 , GRB.MAXIMIZE)
m.update()
m.addConstr(2* x1 +  1* x2 <= 10, "c0")
m.addConstr( 15 * x1 + 30 * x2 <= 200, "c1")
m.addConstr(x1 >= 0)
m.addConstr(x2 >= 0)
m.update()

#用來算枚舉數的限制式

m.addConstr( x2  >= 6, "c2")
'''
m.addConstr( x1  <= 1, "c3")
m.addConstr( x2  >= 6, "c4")
'''
m.setParam( 'OutputFlag', False )

m.optimize()

print('Optimal :')
for v in m.getVars():
    print('%s: %f' % (v.varName, v.x))
print('Obj: %f' % m.objVal)
```

```
Optimal :
x1: 1.333333
x2: 6.000000
Obj: 20.666667
```

設定 X2>=6

```python
from gurobipy import *
m = Model("LP")
'''
x1 = m.addVar( name="x1", vtype=GRB.INTEGER)
x2 = m.addVar( name="x2", vtype=GRB.INTEGER)
'''
x1 = m.addVar( name="x1")
x2 = m.addVar( name="x2")
m.setObjective(2*x1 + 3*x2 , GRB.MAXIMIZE)
m.update()
m.addConstr(2* x1 +  1* x2 <= 10, "c0")
m.addConstr( 15 * x1 + 30 * x2 <= 200, "c1")
m.addConstr(x1 >= 0)
m.addConstr(x2 >= 0)
m.update()

#用來算枚舉數的限制式

m.addConstr( x2  <= 5, "c2")
'''
m.addConstr( x1  <= 1, "c3")
m.addConstr( x2  >= 6, "c4")
'''
m.setParam( 'OutputFlag', False )

m.optimize()

print('Optimal :')
for v in m.getVars():
    print('%s: %f' % (v.varName, v.x))
print('Obj: %f' % m.objVal)
```

```
Optimal :
x1: 2.500000
x2: 5.000000
Obj: 20.000000
```

設定 X2<=5

因為 20 <20.6667 X2<=5 Stop here

向下設定 X1<=1 X1>=2

```
x1 = m.addVar( name="x1")
x2 = m.addVar( name="x2")
m.setObjective(2*x1 + 3*x2 , GRB.MAXIMIZE)
m.update()
m.addConstr(2* x1 +  1* x2 <= 10, "c0")
m.addConstr( 15 * x1 + 30 * x2 <= 200, "c1")
m.addConstr(x1 >= 0)
m.addConstr(x2 >= 0)
m.update()

#用來算枚舉數的限制式

m.addConstr( x2  >= 6, "c2")

m.addConstr( x1  >= 2, "c3")
'''
m.addConstr( x2  >= 6, "c4")
'''
m.setParam( 'OutputFlag', False )

m.optimize()

print('Optimal :')
for v in m.getVars():
    print('%s: %f' % (v.varName, v.x))
print('Obj: %f' % m.objVal)
```

```
AttributeError: Unable to retrieve attribute 'x'
```

X1>=2 not feasible

```
x1 = m.addVar( name="x1")
x2 = m.addVar( name="x2")
m.setObjective(2*x1 + 3*x2 , GRB.MAXIMIZE)
m.update()
m.addConstr(2* x1 +  1* x2 <= 10, "c0")
m.addConstr( 15 * x1 + 30 * x2 <= 200, "c1")
m.addConstr(x1 >= 0)
m.addConstr(x2 >= 0)
m.update()

#用來算枚舉數的限制式

m.addConstr( x2  >= 6, "c2")

m.addConstr( x1  <= 1, "c3")
'''
m.addConstr( x2  >= 6, "c4")
'''
m.setParam( 'OutputFlag', False )

m.optimize()

print('Optimal :')
for v in m.getVars():
    print('%s: %f' % (v.varName, v.x))
print('Obj: %f' % m.objVal)
```

```
Optimal :
x1: 1.000000
x2: 6.166667
Obj: 20.500000
```

向下設定 X2>=7 X2<=6

```
x1 = m.addVar( name="x1")
x2 = m.addVar( name="x2")
m.setObjective(2*x1 + 3*x2 , GRB.MAXIMIZE)
m.update()
m.addConstr(2* x1 +  1* x2 <= 10, "c0")
m.addConstr( 15 * x1 + 30 * x2 <= 200, "c1")
m.addConstr(x1 >= 0)
m.addConstr(x2 >= 0)
m.update()

#用來算枚舉數的限制式

m.addConstr( x2  >= 6, "c2")
m.addConstr( x1  <= 1, "c3")
m.addConstr( x2  >= 7, "c4")
m.setParam( 'OutputFlag', False )

m.optimize()

print('Optimal :')
for v in m.getVars():
    print('%s: %f' % (v.varName, v.x))
print('Obj: %f' % m.objVal)
```

```
AttributeError: Unable to retrieve attribute 'x'
```

X2>=7 not feasible

```
x1 = m.addVar( name="x1")
x2 = m.addVar( name="x2")
m.setObjective(2*x1 + 3*x2 , GRB.MAXIMIZE)
m.update()
m.addConstr(2* x1 +  1* x2 <= 10, "c0")
m.addConstr( 15 * x1 + 30 * x2 <= 200, "c1")
m.addConstr(x1 >= 0)
m.addConstr(x2 >= 0)
m.update()

#用來算枚舉數的限制式

m.addConstr( x2  >= 6, "c2")
m.addConstr( x1  <= 1, "c3")
m.addConstr( x2  <= 6, "c4")
m.setParam( 'OutputFlag', False )

m.optimize()

print('Optimal :')
for v in m.getVars():
    print('%s: %f' % (v.varName, v.x))
print('Obj: %f' % m.objVal)
```

```
Optimal :
x1: 1.000000
x2: 6.000000
Obj: 20.000000
```

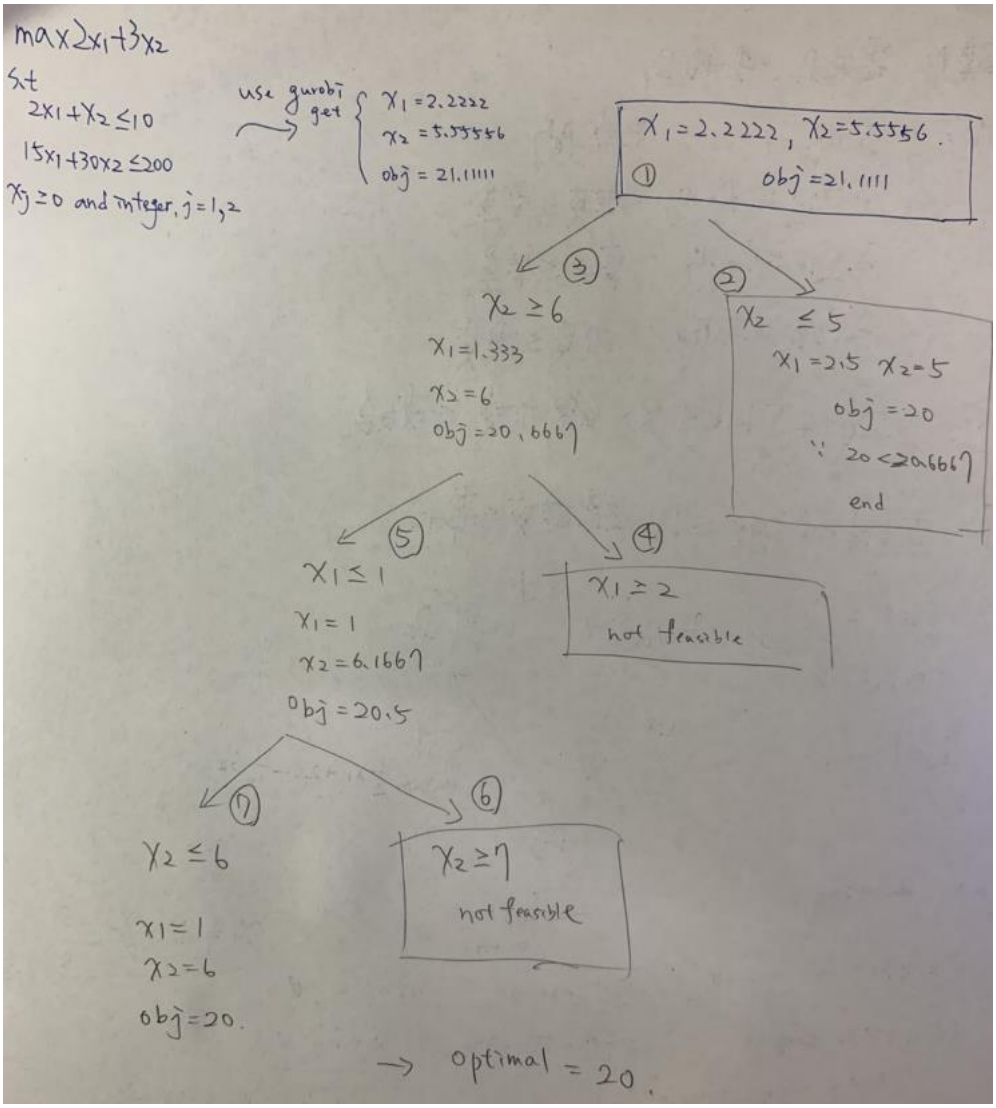
最後得最佳解 x1 =1 x2=6 obj = 20

最後用 gurobi 解 integer programming 來驗算答案

```
from gurobipy import *
m = Model("LP")
x1 = m.addVar( name="x1", vtype=GRB.INTEGER)
x2 = m.addVar( name="x2", vtype=GRB.INTEGER)
'''
x1 = m.addVar( name="x1")
x2 = m.addVar( name="x2")
'''
m.setObjective(2*x1 + 3*x2 , GRB.MAXIMIZE)
m.update()
m.addConstr(2* x1 +  1* x2 <= 10, "c0")
m.addConstr( 15 * x1 + 30 * x2 <= 200, "c1")
m.addConstr(x1 >= 0)
m.addConstr(x2 >= 0)
m.update()

#用來算枚舉數的限制式
'''
m.addConstr( x2  >= 6, "c2")
m.addConstr( x1  <= 1, "c3")
m.addConstr( x2  <= 6, "c4")
'''
m.setParam( 'OutputFlag', False )
m.optimize()
print('Optimal :')
for v in m.getVars():
    print('%s: %f' % (v.varName, v.x))
print('Obj: %f' % m.objVal)
```

```
Optimal :
x1: 1.000000
x2: 6.000000
Obj: 20.000000
```


max 2x1+3x2
s.t
2x1+x2 ≤10
15x1+30x2 ≤200
xj ≥0 and integer, j=1,2
use gurobi get
x1=2.2222
x2=5.5556
obj=21.1111
x1=2.2222, x2=5.5556.
① obj=21.1111
③
x2 ≥6
x1=1.333
x2=6
obj=20.6667
②
x2 ≤5
x1=2.5 x2=5
obj=20
∵ 20<20.6667
end
⑤
x1 ≤1
x1=1
x2=6.1667
obj=20.5
④
x1 ≥2
not feasible
⑦
x2 ≤6
x1=1
x2=6
obj=20.
⑥
x2 ≥7
not feasible
→ optimal = 20.