INDR 220: Introduction to Computing for Operations Research
Homework 3: The Set Covering Problem
Deadline: December 5, 2022, 11:59 PM

In this homework, you will implement a Python script that solves the set covering problem using CPLEX. As a specific example, consider the scheduling of airline flight personnel. The airline has M "legs" to be flown. The airline must schedule its K crews on N routes to cover these flights. One crew, for example, might be scheduled to fly a route containing two legs just mentioned. The decision variables, then, specify the scheduling of the crews to routes:

$$x_j = \begin{cases} 1 & \text{if a crew is assigned to route } j, \\ 0 & \text{otherwise.} \end{cases}$$

Let

$$a_{ij} = \begin{cases} 1 & \text{if leg } i \text{ is included on route } j, \\ 0 & \text{otherwise,} \end{cases}$$

and c_j is the cost for assigning a crew to route j.

The coefficients a_{ij} define the acceptable combinations of legs and routes, taking into account such characteristics as sequencing of legs for making connections between flights and for including in the routes ground time for maintenance. The model becomes:

$$\begin{aligned} \text{minimize} \quad & z = \sum_{j=1}^{N} c_j x_j \\ \text{subject to:} \quad & \sum_{j=1}^{N} a_{ij} x_j \geq 1 \quad i = 1, 2, \ldots, M \\ & \sum_{j=1}^{N} x_j = K \\ & x_j \in \{0, 1\} \quad j = 1, 2, \ldots, N. \end{aligned}$$

An example of a set covering problem with eleven legs, twelve flights, and three crews is given as

	Routes											
Legs	1	2	3	4	5	6	7	8	9	10	11	12
1.SF to LA	1			1			1			1		
2.SF to Denver		1			1			1			1	
3.SF to Seattle			1			1			1			1
4.LA to Chicago				1			1		1	1		1
5.LA to SF	1					1				1	1	
6.Chicago to Denver				1	1				1			
7.Chicago to Seattle							1	1		1	1	1
8.Denver to SF		1		1	1				1			
9.Denver to Chicago					1			1			1	
10.Seattle to SF			1				1	1				1
11.Seattle to LA						1			1	1	1	1
Cost (in $1000's)	2	3	4	6	7	5	7	8	9	9	8	9

$$
\begin{aligned}
\text{minimize} \quad & z = +2000x_1 + 3000x_2 + 4000x_3 + 6000x_4 + 7000x_5 + 5000x_6 \\
& \qquad + 7000x_7 + 8000x_8 + 9000x_9 + 9000x_{10} + 8000x_{11} + 9000x_{12} \\
\text{subject to:} \quad & + x_1 + x_4 + x_7 + x_{10} \geq 1 \\
& + x_2 + x_5 + x_8 + x_{11} \geq 1 \\
& + x_3 + x_6 + x_9 + x_{12} \geq 1 \\
& + x_4 + x_7 + x_9 + x_{10} + x_{12} \geq 1 \\
& + x_1 + x_6 + x_{10} + x_{11} \geq 1 \\
& + x_4 + x_5 + x_9 \geq 1 \\
& + x_7 + x_8 + x_{10} + x_{11} + x_{12} \geq 1 \\
& + x_2 + x_4 + x_5 + x_9 \geq 1 \\
& + x_5 + x_8 + x_{11} \geq 1 \\
& + x_3 + x_7 + x_8 + x_{12} \geq 1 \\
& + x_6 + x_9 + x_{10} + x_{11} + x_{12} \geq 1 \\
& + x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} = 3 \\
& x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12} \in \{0, 1\}.
\end{aligned}
$$

This problem will be represented using two `.txt` files, namely, `flights.txt`, `costs.txt` and the crew count parameter K, which is equal to 3 for the example problem. The first file contains the leg-route matchings (i.e., where a_{ij}'s are nonzero), and it is composed of the following lines for the example problem:

```
flights.txt
-----------
1 1
1 4
1 7
1 10
2 2
2 5
2 8
2 11
3 3
3 6
3 9
3 12
4 4
4 7
4 9
4 10
4 12
5 1
5 6
5 10
5 11
6 4
6 5
```

```
6 9
7 7
7 8
7 10
7 11
7 12
8 2
8 4
8 5
8 9
9 5
9 8
9 11
10 3
10 7
10 8
10 12
11 6
11 9
11 10
11 11
11 12
```

The second file contains the route costs (i.e., c_j), and it is composed of the following lines for the example problem:

```
costs.txt
---------
1 2000
2 3000
3 4000
4 6000
5 7000
6 5000
7 7000
8 8000
9 9000
10 9000
11 8000
12 9000
```

One optimum solution of the example problem is as follows:

$$x_1^\star = x_2^\star = x_5^\star = x_6^\star = x_7^\star = x_8^\star = x_9^\star = x_{10}^\star = x_{12}^\star = 0$$
$$x_3^\star = x_4^\star = x_{11}^\star = 1$$
$$z^\star = 18000.$$

Another optimum solution of the example problem is as follows:

$$x_1^\star = x_5^\star = x_{12}^\star = 1$$
$$x_2^\star = x_3^\star = x_4^\star = x_6^\star = x_7^\star = x_8^\star = x_9^\star = x_{10}^\star = x_{11}^\star = 0$$
$$z^\star = 18000.$$

Implement your algorithm to solve the set covering problem in a single interactive Python notebook using Azure Lab Services. Your notebook should include at least the following function definition that takes the file paths of two input files and the crew count as parameters and returns the solution found.

```
def set_covering_problem(flights_file, costs_file, K):
    #implement your algorithm here
    return(x_star, obj_star)
```

What to submit: You need to submit your source code in a single file (`.py` file that you will download from Azure Lab Services by following "File" / "Download as" / "Python (.py)" menu items) named as `STUDENTID.py`, where `STUDENTID` should be replaced with your 7-digit student number.

How to submit: Submit the file you created to Blackboard. Please follow the exact style mentioned and do not send a file named as `STUDENTID.py`. Submissions that do not follow these guidelines will not be graded.

Late submission policy: Late submissions will not be graded.

Cheating policy: Very similar submissions will not be graded.
