COMP 5413

Topics in Smart Health Informatics

Assignment 1

Khushal Paresh Thaker

1106937

13 July 2020

Part 1

	P1	P2
D4	130	95
D5	118	83

Considering,

	P1	P2
D4	X_{41}	X_{42}
D5	X_{51}	X_{52}

Min Z = $130X_{41} + 95X_{42} + 118X_{51} + 83X_{52}$

Such that,

Doctor's Constraints –

$X_{41} + X_{42} = 1$

$X_{51} + X_{52} = 1$

Patient's Constraints –

$X_{14} + X_{24} = 1$

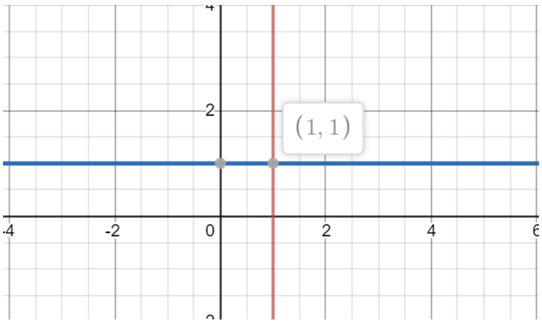
$X_{15} + X_{25} = 1$

X_{ij} = {1 if I is allocated to j}
or
{0, otherwise}

Minimize,

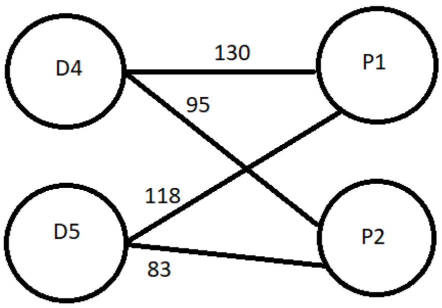
$\sum\sum C_{ij} X_{ij}$
$\sum X_{ij} = 1\ \forall i$
$\sum X_{ij} = 1\ \forall j$
$X_{i,j} = 0,1$

Drawing a graph for an equation with type X + Y = 1, we get either X = (0,1) and Y = (0,1) and lines intersecting at 1,1.



And the region surrounding (1,1) would be the solution ideally. But, Having 4 different variables and 4 equations, it becomes difficult to represent this via a graph.

This causes an issue as this provides multiple solutions. When we take the optimal value, it is same for both the solutions, 213. We get 213 by adding (118 + 95) or (130 + 83).



Considering the values provided and selecting the lowest value first, we can find that lowest time is 83 which is taken by D5 to diagnose P2. So, we can assign this accordingly and then D4 would be assigned to P1.

D4 -> P1

D5 -> P2

Optimal Value is 213 which is also shown using the gurobi output (part 3).

Part 2

Filling up the matrix with random values,

	P1	P2	P3	P4	P5	P6
D1	75	140	50	60	120	110
D2	90	55	70	100	110	65
D3	95	70	105	130	140	70
D4	130	95	75	85	80	90
D5	118	83	120	60	115	95
D6	110	90	125	95	85	60

Considering,

Constraints	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**
D1	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	X_{16}
D2	X_{21}	X_{22}	X_{23}	X_{24}	X_{25}	X_{26}
D3	X_{31}	X_{32}	X_{33}	X_{34}	X_{35}	X_{36}
D4	X_{41}	X_{42}	X_{43}	X_{44}	X_{45}	X_{46}
D5	X_{51}	X_{52}	X_{53}	X_{54}	X_{55}	X_{56}
D6	X_{61}	X_{62}	X_{63}	X_{64}	X_{65}	X_{66}

Min Z = $70X_{11} + 60X_{12} + 80X_{13} + 80X_{14} + 90X_{15} + 120X_{16} +$

$55X_{21} + 70X_{22} + 90X_{23} + 75X_{24} + 100X_{25} + 90X_{26} +$

$65X_{31} + 60X_{32} + 65X_{33} + 55X_{34} + 85X_{35} + 70X_{36} +$

$130X_{41} + 95X_{42} + 70X_{43} + 85X_{44} + 125X_{45} + 110X_{46} +$

$118X_{51} + 83X_{52} + 120X_{53} + 90X_{54} + 55X_{55} + 75X_{56} +$

$110X_{61} + 95X_{66} + 70X_{63} + 110X_{64} + 130X_{65} + 60X_{66}$

Such that,

Doctor's Constraints -

$X_{11} + X_{12} + X_{13} + X_{14} + X_{15} + X_{16} = 1$

$X_{21} + X_{22} + X_{23} + X_{24} + X_{25} + X_{26} = 1$

$X_{31} + X_{32} + X_{33} + X_{34} + X_{35} + X_{36} = 1$

$X_{41} + X_{42} + X_{43} + X_{44} + X_{45} + X_{46} = 1$

$X_{51} + X_{52} + X_{53} + X_{54} + X_{55} + X_{56} = 1$

$X_{61} + X_{66} + X_{63} + X_{64} + X_{65} + X_{66} = 1$

Patient's Constraints –

$X_{11} + X_{21} + X_{31} + X_{41} + X_{51} + X_{61} = 1$

$X_{12} + X_{22} + X_{32} + X_{42} + X_{52} + X_{62} = 1$

$X_{13} + X_{23} + X_{33} + X_{43} + X_{53} + X_{63} = 1$

$X_{14} + X_{24} + X_{34} + X_{44} + X_{54} + X_{64} = 1$

$X_{15} + X_{25} + X_{35} + X_{45} + X_{55} + X_{65} = 1$

$X_{16} + X_{26} + X36 + X_{46} + X_{56} + X_{66} = 1$

$X_{ij} =$ {1 if I is allocated to j}
or
{0, otherwise}

Minimize,

$\sum\sum C_{ij} X_{ij}$
$\sum X_{ij} = 1 \ \forall i$
$\sum X_{ij} = 1 \ \forall j$
$X_{i,j} = 0,1$

Considering the following matrix,

	P1	**P2**	**P3**	**P4**	**P5**	**P6**
D1	75	140	50	60	120	110
D2	90	55	70	100	110	65
D3	95	70	105	130	140	70
D4	130	95	75	85	80	90

D5	118	83	120	60	115	95
D6	110	90	125	95	85	60

Using Hungarian approach, we ***subtract the row minima,***

25	90	0	10	70	60	(-50)
35	0	15	45	55	10	(-55)
25	0	35	60	70	0	(-70)
55	20	0	10	5	15	(-75)
58	23	60	0	55	35	(-60)
50	30	65	35	25	0	(-60)

Subtracting the column minima,

0	90	0	10	65	60
10	0	15	45	50	10
0	0	35	60	65	0
30	20	0	10	0	15
33	23	60	0	50	35
25	30	65	35	20	0
(-25)				(-5)	

The *optimal solutions are with minimum number of lines covering the zeroes*. And this satisfies the condition for an optimal assignment.

0	90	**0**	10	65	60
10	**0**	15	45	50	10
0	0	35	60	65	0
30	20	0	10	**0**	15
33	23	60	**0**	50	35
25	30	65	35	20	**0**

Optimal Assignment

75	140	**50**	60	120	110
90	**55**	70	100	110	65
95	70	105	130	140	70
130	95	75	85	**80**	90
118	83	120	**60**	115	95
110	90	125	95	85	**60**

D1 -> P3
D2 -> P2
D3 -> P1
D4 -> P5
D5 -> P4
D6 -> P6

The optimal value equals 400 which is also produced by the gurobi code (part 3).

Part 3

Solution for Part 1 via Gurobi code:

In [77]:
```
# running the optimization engine
m_part1.optimize()
```

```
Gurobi Optimizer version 9.0.2 build v9.0.2rc0 (win64)
Optimize a model with 4 rows, 4 columns and 8 nonzeros
Model fingerprint: 0x558e7b16
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [8e+01, 1e+02]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+00, 1e+00]
Presolve removed 4 rows and 4 columns
Presolve time: 0.02s
Presolve: All rows and columns removed
Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    2.1300000e+02   0.000000e+00   0.000000e+00      0s

Solved in 0 iterations and 0.03 seconds
Optimal objective  2.130000000e+02
```

In [79]:
```
#Displaying the optimal values of decision variables
for v in m_part1.getVars():
    if (abs(v.x) > 1e-6):
        print(v.varName, v.x)

# displaying the total minutes taken
print('Total Minutes taken', m_part1.objVal)
```

```
assign[D4,P1] 1.0
assign[D5,P2] 1.0
Total Minutes taken 213.0
```

Solution for Part 2 via Gurobi code:

In [19]:
```
# running the optimization engine
m.optimize()
```

```
Gurobi Optimizer version 9.0.2 build v9.0.2rc0 (win64)
Optimize a model with 12 rows, 36 columns and 72 nonzeros
Model fingerprint: 0x62246923
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [5e+01, 1e+03]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+00, 1e+00]
Presolve time: 0.02s
Presolved: 12 rows, 36 columns, 72 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    3.8000000e+02   2.000000e+00   0.000000e+00      0s
       2    4.0000000e+02   0.000000e+00   0.000000e+00      0s

Solved in 2 iterations and 0.03 seconds
Optimal objective  4.000000000e+02
```

In [20]:
```
#Displaying the optimal values of decision variables
for v in m.getVars():
    if (abs(v.x) > 1e-6):
        print(v.varName, v.x)

# displaying the total minutes taken
print('Total Minutes taken', m.objVal)
```

```
assign[D1,P3] 1.0
assign[D2,P2] 1.0
assign[D3,P1] 1.0
assign[D4,P5] 1.0
assign[D5,P4] 1.0
assign[D6,P6] 1.0
Total Minutes taken 400.0
```