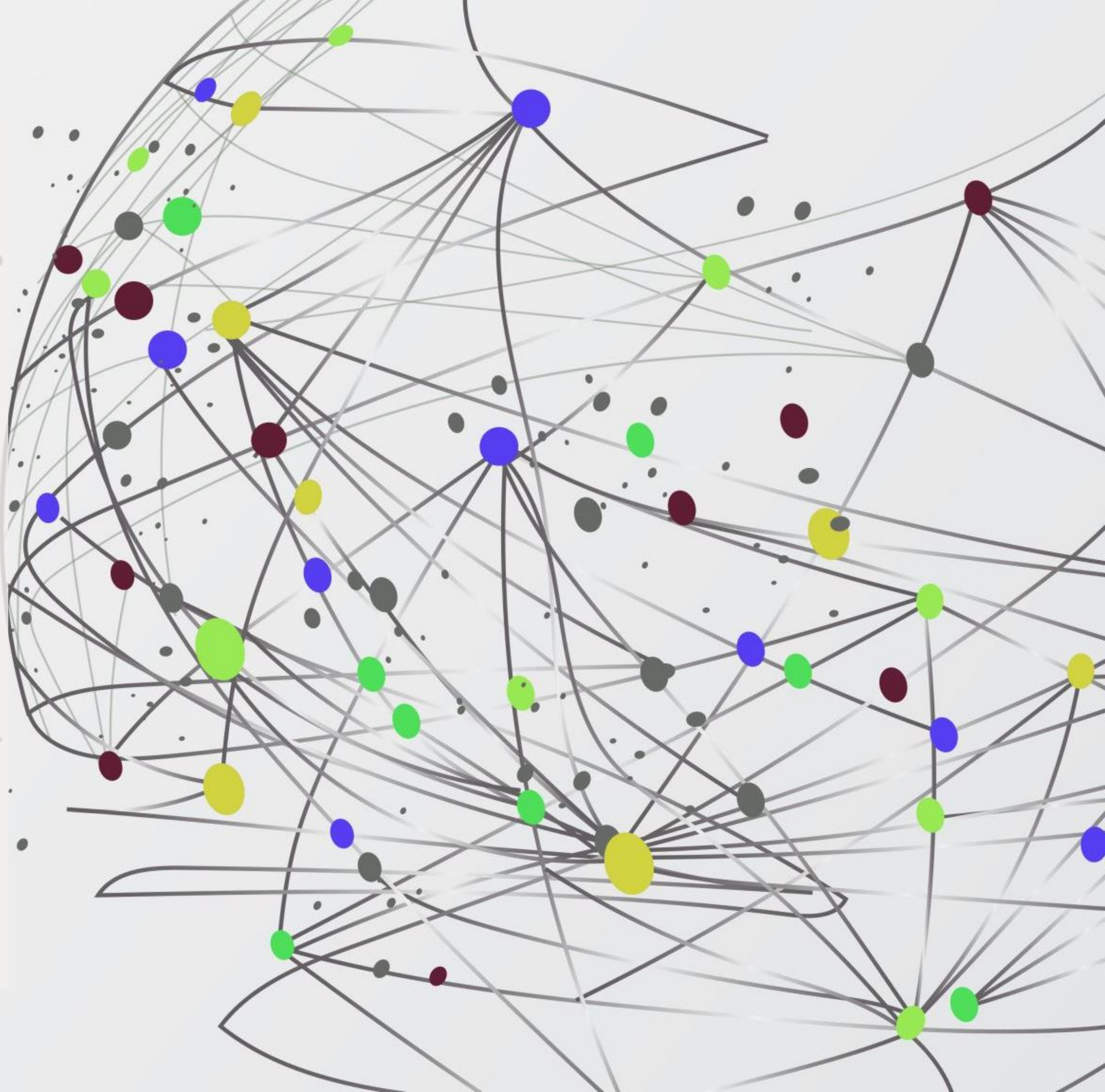
GRAPH COLOURING PROBLEM

OPTIMIZATION METHODS - II

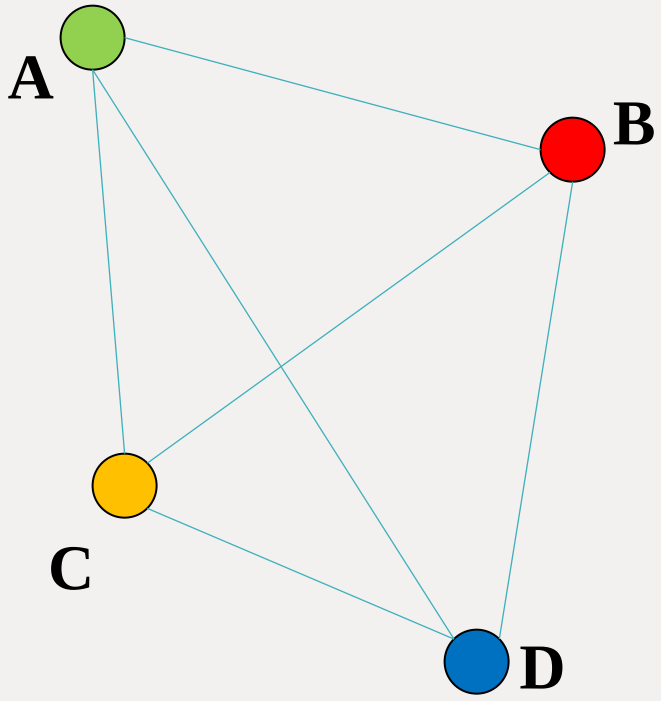
-Dikshant Joshi



Agenda

- Overview
- Application of Graph Coloring Problem
- Problem Statement
- Algebraic Formulation
- Modelling using Python
- Modelling using Excel
- Results

OVERVIEW



- The graph coloring problem is a well-known problem in computer science and graph theory.
- It is the problem of assigning colors to the vertices of a graph such that no two adjacent vertices have the same color, while using the fewest number of colors possible

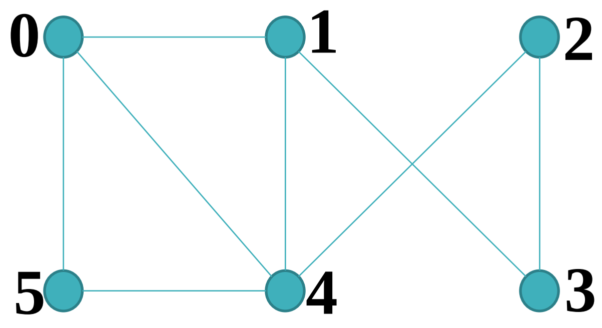
APPLICATION OF GCP

- Timetabling
- Register allocation
- Airline Scheduling

Note: Graph coloring problem is NP Hard .This means that there is no known algorithm that can solve it in polynomial time for all instances

PROBLEM STATEMENT

- There are 6 nodes in a graph given and we need to colour these node in a way that adjacent nodes do not have same colour.
- The adjacency matrix is shown in the below graph which tells if two nodes are connected (1) or not(0).
- Based on the given problem we need to find optimum number of colors required to color the nodes of the graph



	0	1	2	3	4	5
0	0	1	0	0	1	1
1	1	0	0	1	1	0
2	0	0	0	1	1	0
3	0	1	1	0	0	0
4	1	1	1	0	0	1
5	1	0	0	0	1	0

ALGEBRAIC FORMULATION

Yj

0	0	0	0	0	0

- **Decision Variables**
  - Yj : a binary variable that indicates whether color j has been used or not (1 if used , 0 otherwise)
  - Xij : a binary variable that indicates whether node i is colored with color j or not

Xij

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

- **Objective Function:**
  - Minimize Z: $\sum_i Y_i$ $i \in$ N,

Where N is the set of all nodes.

ALGEBRAIC FORMULATION

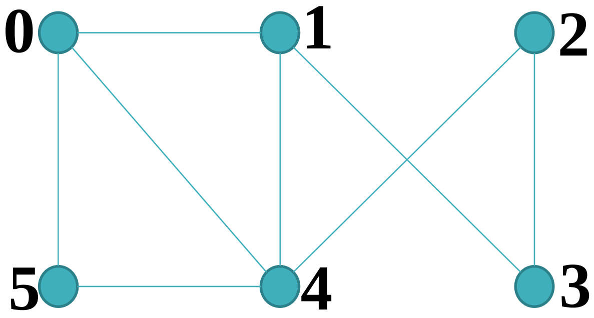
- **Constraints**
  - Each node should have only one color:

    $\sum_j X_{ij}$ = 1 for all i

ALGEBRAIC FORMULATION

- **Constraints**
  - Each node should have only one color:

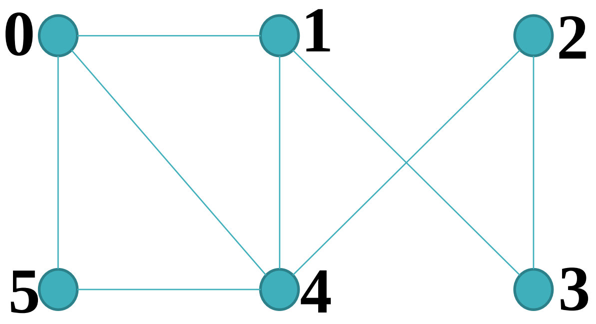
    $\sum_j X_{ij}$ = 1 for all i



ALGEBRAIC FORMULATION

- **Constraints**
  - Each node should have only one color:

    $\sum_j X_{ij}$ = 1 for all i

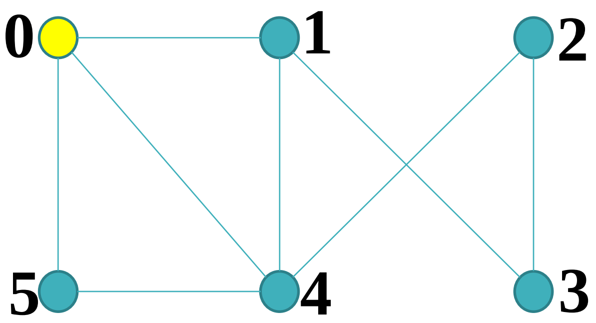


	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

ALGEBRAIC FORMULATION

- Constraints
  - Each node should have only one color:

    $\sum_j X_{ij}$ = 1 for all j



	0	1	2	3	4	5
0	1	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

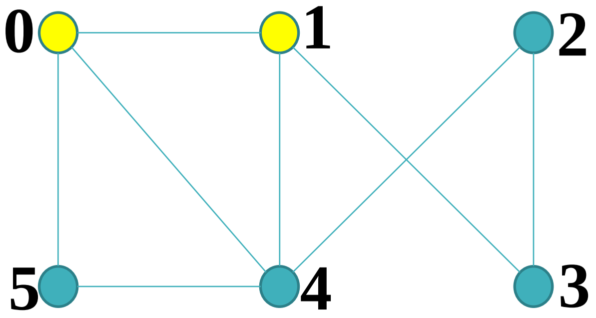
ALGEBRAIC FORMULATION

- Constraints
  - Each node should have only one color:

    $\sum_j X_{ij}$ = 1 for all j

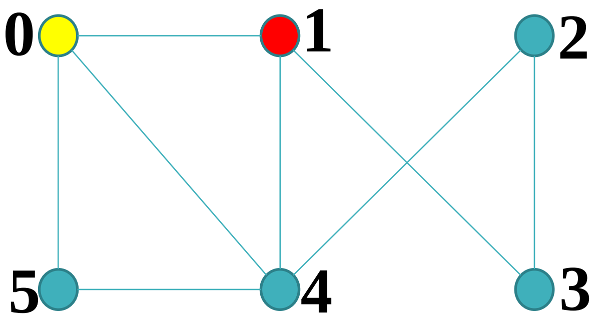
  - Adjacent nodes must have different color:

    $X_{ik} + X_{jk}$ <= 1 for all A_{ij} = 1 , k ∈ N where A is adjacency matrix



	0	1	2	3	4	5
0	1	0	0	0	0	0
1	1	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

ALGEBRAIC FORMULATION



- **Constraints**
  - Each node should have only one color:

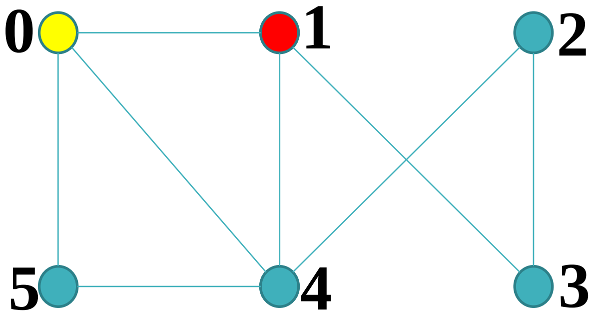
    $\sum_j X_{ij}$ = 1 for all j

  - Adjacent nodes must have different color:

    $X_{ik} + X_{jk}$ <= 1 for all A_{ij} = 1 , k ∈ N where A is adjacency matrix

	0	1	2	3	4	5
0	1	0	0	0	0	0
1	0	1	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

ALGEBRAIC FORMULATION



- **Constraints**
  - Each node should have only one color:

    $\sum_j X_{ij} = 1$ for all j

  - Adjacent nodes must have different color:

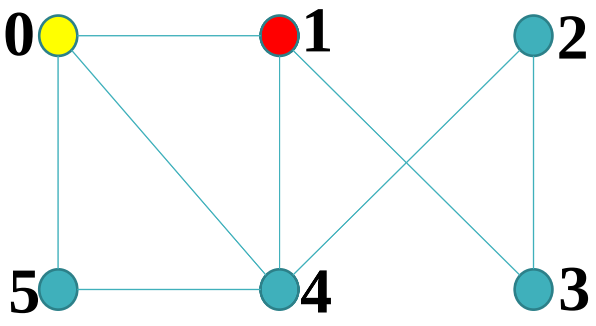
    $X_{ik} + X_{jk} <= 1$ for all $A_{ij} = 1$, $k \in N$ where A is adjacency matrix

  - The colour should be used only n number of times:

    $\sum_i X_{ij} <= n * Y_j$ for all j

	0	1	2	3	4	5
0	1	0	0	0	0	0
1	0	1	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

ALGEBRAIC FORMULATION



- **Constraints**
  - Each node should have only one color:

    $\sum_j X_{ij} = 1$ for all j

  - Adjacent nodes must have different color:

    $X_{ik} + X_{jk} <= 1$ for all $A_{ij} = 1$, $k \in N$ where A is adjacency matrix

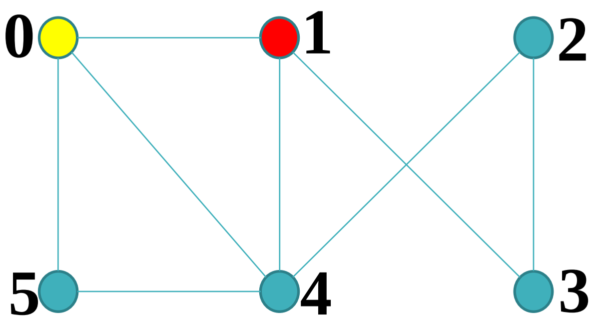
  - The colour should be used only n number of times:

    $\sum_i X_{ij} <= n * Y_j$ for all j

0	0	0	0	0	0

	0	1	2	3	4	5
0	1	0	0	0	0	0
1	0	1	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

ALGEBRAIC FORMULATION



- **Constraints**
  - Each node should have only one color:

    $\sum_j X_{ij} = 1$ for all j

  - Adjacent nodes must have different color:

    $X_{ik} + X_{jk} <= 1$ for all $A_{ij} = 1$, $k \in N$ where A is adjacency matrix

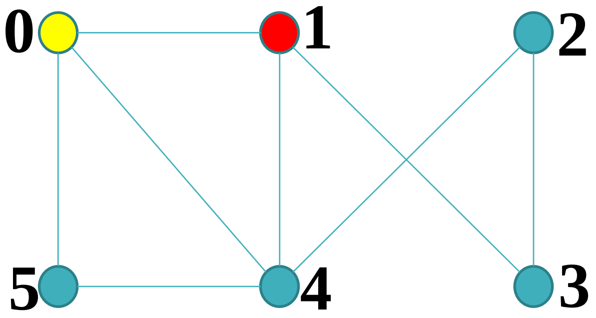
  - The colour should be used only n number of times:

    $\sum_i X_{ij} <= n * Y_j$ for all j

1	0	0	0	0	0

	0	1	2	3	4	5
0	1	0	0	0	0	0
1	0	1	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

ALGEBRAIC FORMULATION



- **Constraints**
  - Each node should have only one color:

    $\sum_j X_{ij} = 1$ for all j

  - Adjacent nodes must have different color:

    $X_{ik} + X_{jk} <= 1$ for all $A_{ij} = 1$, $k \in N$ where A is adjacency matrix

  - The colour should be used only n number of times:

    $\sum_i X_{ij} <= n * Y_i$ for all i

	0	1	2	3	4	5
0	1	0	0	0	0	0
1	0	1	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

1	1	0	0	0	0

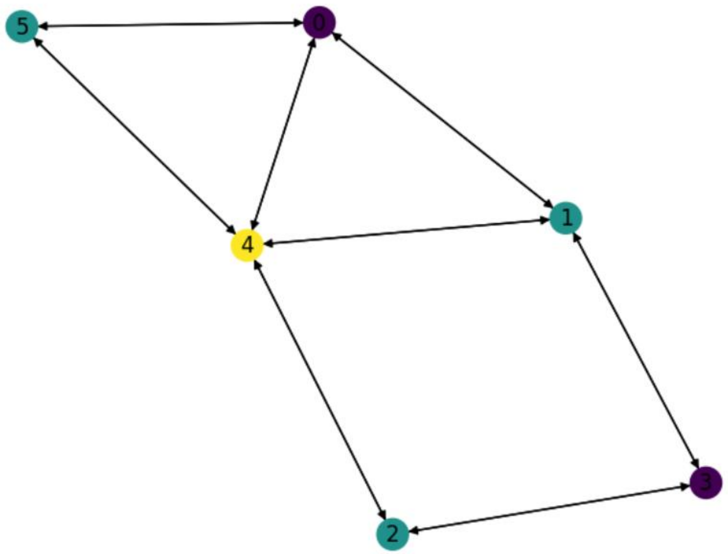
MODELLING USING PYTHON

```python
# Define the adjacency matrix for the graph with 5 nodes
adj_matrix = np.array([[0, 1, 0, 0, 1, 1],
                       [1, 0, 0, 1, 1, 0],
                       [0, 0, 0, 1, 1, 0],
                       [0, 1, 1, 0, 0, 0],
                       [1, 1, 1, 0, 0, 1],
                       [1, 0, 0, 0, 1, 0]])

# Define the number of nodes
num_nodes = 6

# Define the decision variable
#Boolean variables y[j] that colour j is used or not
#Assumption: we have n number of colours available where n = nodes
y = cp.Variable(num_nodes, boolean=True)

#Boolean variable array X[i,j] that node i is coloured with colour j or not
x = cp.Variable((6,num_nodes), integer=True)

# Define the objective function
obj = cp.Minimize(cp.sum(y))

# Define the constraints

#This allows to have only one color per node
constraints = []
for i in range(num_nodes):
    for j in range(num_nodes):
        constraints.append(x[i, j] >= 0)
    constraints.append(cp.sum(x[i, :]) == 1)

#This allows the adjacent node to have different colors
for i in range(num_nodes):
    for j in range(num_nodes):
        if adj_matrix[i, j] == 1:
            for k in range(num_nodes):
                constraints.append(x[i, k] + x[j, k] <= 1)

#This constraint is indirectly ensuring that if a colour is used to colour a node than it should reflect in the array y.
constraints.append(cp.sum(x, axis=0) <= num_nodes * y)
```

```python
# Solve the problem
problem = cp.Problem(obj, constraints)
problem.solve(solver=cp.GLPK_MI)

# Print the solution
print("Minimum number of colors:", int(obj.value))
print("Node colors:")
print(np.argmax(x.value, axis=1))
```

RESULT

```
Minimum number of colors: 3
Node colors:
[2 3 3 2 4 3]
```



The Optimum Number of colours for this Graph colouring problem is : 3
With colours as:

0	1	2	3	4	5
2	3	3	2	4	3

MODELLING USING EXCEL

	Nodes	0	1	2	3	4	5			
	Color	2	0	0	1	1	0			
	Max colors	6	6	6	6	6	6			
									Minimize sum	4
Constraints										
	Each adjacent node should have different colour		1st Node	2nd Node	Color of 1st node	Color of 2nd node	ABS(Color of 2nd - Color of 1st)	>=	1	
	Node 0	Node 1	0	1	0	2	2	>=	1	
		Node 4	0	4	1	2	1	>=	1	
		Node 5	0	5	0	2	2	>=	1	
	Node 1	Node 0	1	0	2	0	2	>=	1	
		Node 4	1	4	1	0	1	>=	1	
		Node 3	1	3	1	0	1	>=	1	
	Node 2	Node 3	2	3	1	0	1	>=	1	
		Node 4	2	4	1	0	1	>=	1	
	Node 3	Node 2	3	2	0	1	1	>=	1	
		Node 1	3	1	0	1	1	>=	1	
	Node 4	Node 1	4	1	0	1	1	>=	1	
		Node 2	4	2	0	1	1	>=	1	
		Node 0	4	0	2	1	1	>=	1	
		Node 5	4	5	0	1	1	>=	1	
	Node 5	Node 0	5	0	2	0	2	>=	1	
		Node 4	5	4	1	0	1	>=	1	
				Y matrix						
Colour	0	1	2	3	4	5	6		No. of colors used	
Used or not	1	1	1	0	0	0	0		3	

Set Objective: $K$6

To: Max Min Value Of: 0

By Changing Variable Cells:

$C$3:$H$3

Subject to the Constraints:

$C$3:$H$3 <= $C$4:$H$4
$C$3:$H$3 = integer
$H$9:$H$24 >= $J$9:$J$24

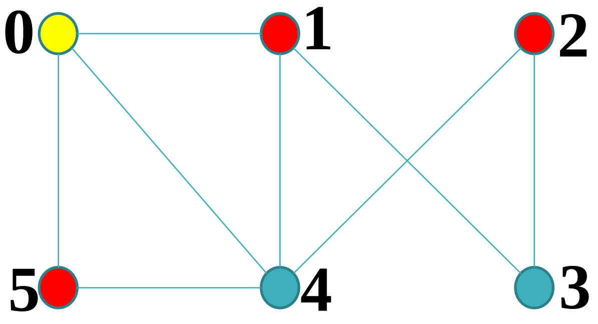
Make Unconstrained Variables Non-Negative

Select a Solving Method: GRG Nonlinear

RESULT

Nodes	0	1	2	3	4	5
Color	2	0	0	1	1	0

No. of colors used
3



The Optimum Number of colours for this Graph colouring problem is : 3
With colours as:

0	1	2	3	4	5
2	0	0	1	1	0

THANK YOU

QUESTIONS ?