GRAPH COLORING

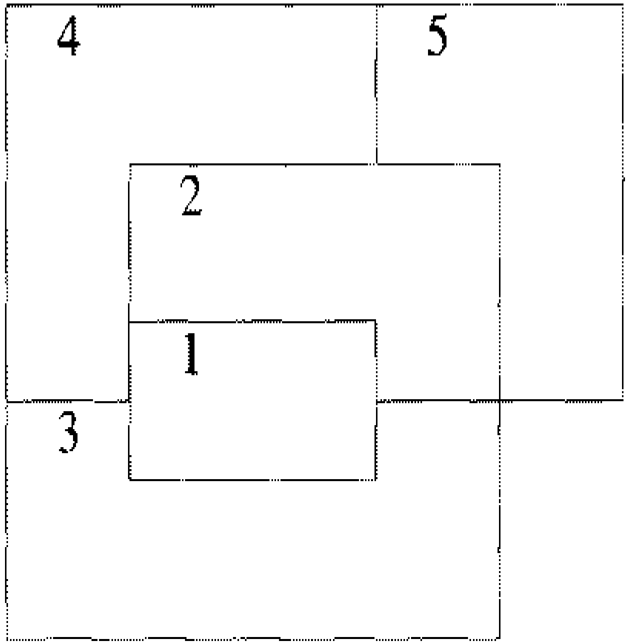
- Let ‘G’ be a graph and ‘m’ be a given positive integer. If the nodes of ‘G’ can be colored in such a way that no two adjacent nodes have the same color. Yet only ‘M’ colors are used. So it’s called M-color ability decision problem.

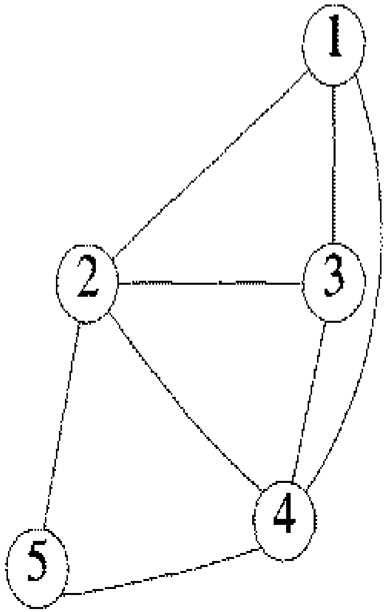
- The graph G can be colored using the smallest integer ‘m’. This integer is referred to as chromatic number of the graph.

- A graph is said to be planar iff it can be drawn on plane in such a way that no two edges cross each other.

- Suppose we are given a map then, we have to convert it into planar. Consider each and every region as a node. If two regions are adjacent then the corresponding nodes are joined by an edge.

A map and its planar graph representation





STEPS TO COLOR THE GRAPH:

- First create the adjacency matrix graph(1:m,1:n) for a graph, if there is an edge between i,j then C(i,j) = 1 otherwise C(i,j) =0.

- The Colors will be represented by the integers 1,2,…..m and the solutions will be stored in the array X(1),X(2),………..,X(n) ,X(index) is the color, index is the node.

- He formula is used to set the color is,
  X(k) = (X(k)+1) % (m+1)

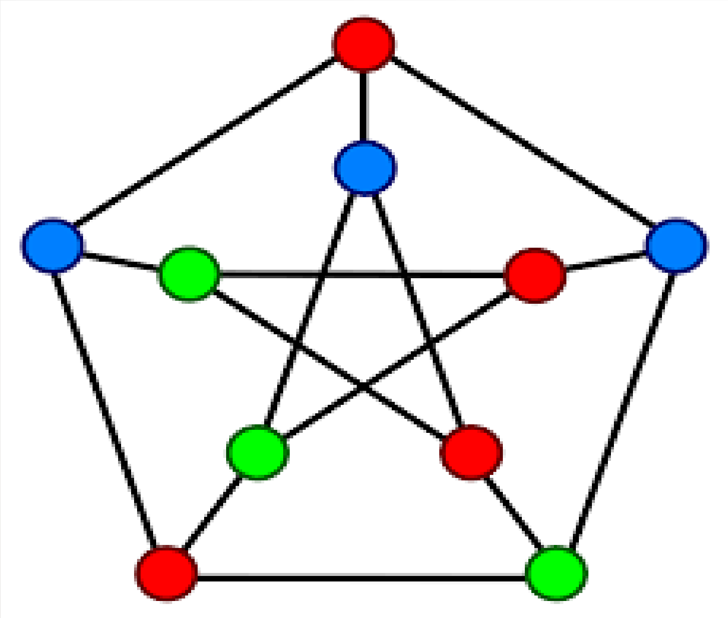
- First one chromatic number is assigned ,after assigning a number for 'k' node, we have to check whether the adjacent nodes has got the same values if so then we have to assign the next value.

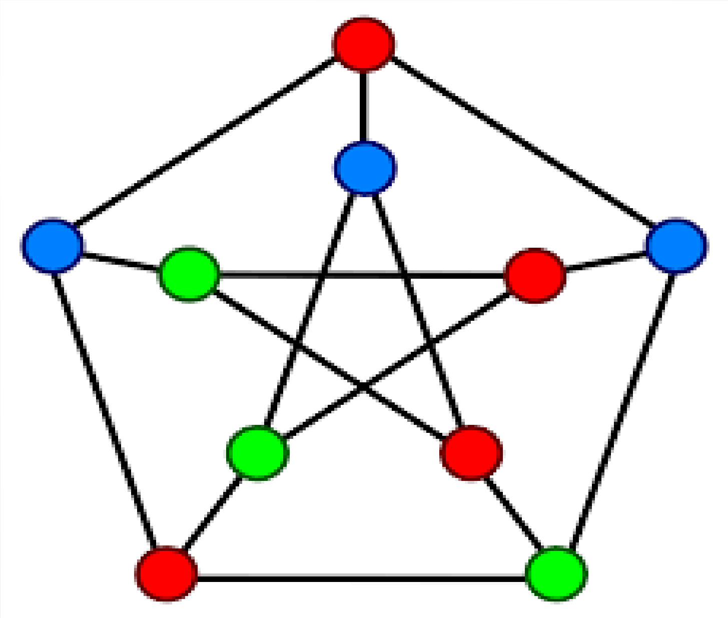
- Repeat the procedure until all possible combinations of colors are found.

- The function which is used to check the adjacent nodes and same color is,

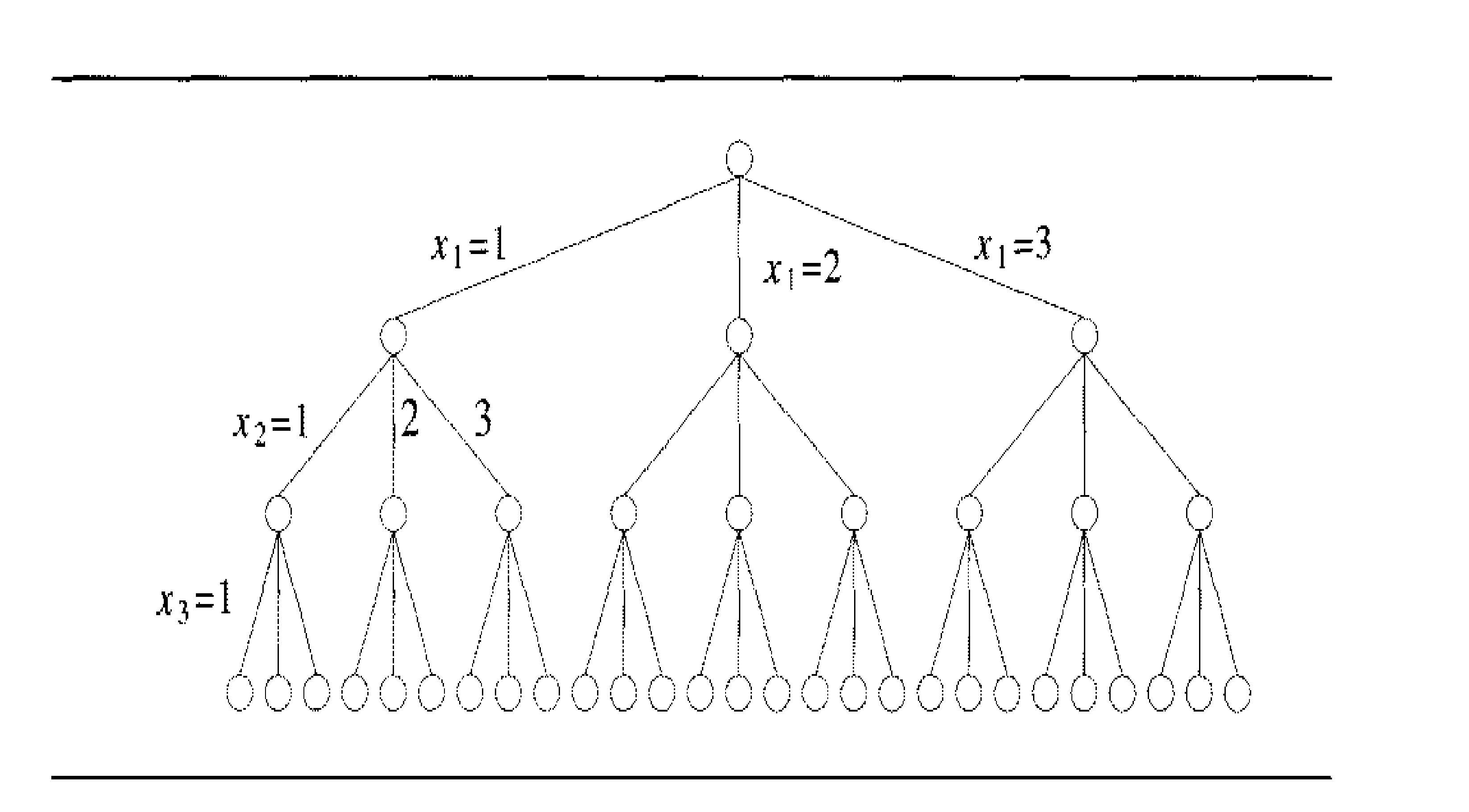
If((Graph (k,j) == 1) and X(k) = X(j))

GRAPH COLORING EXAMPLE

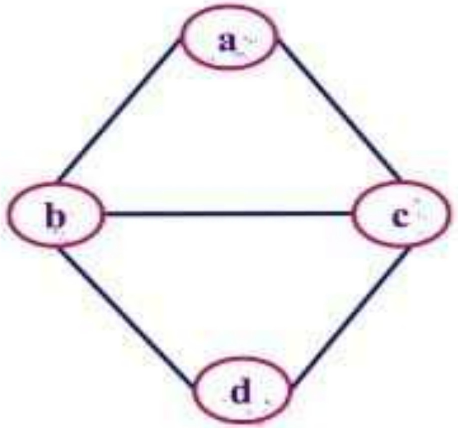




STATE SPACE TREE FOR MCOLORING WHEN N=3 AND M=3



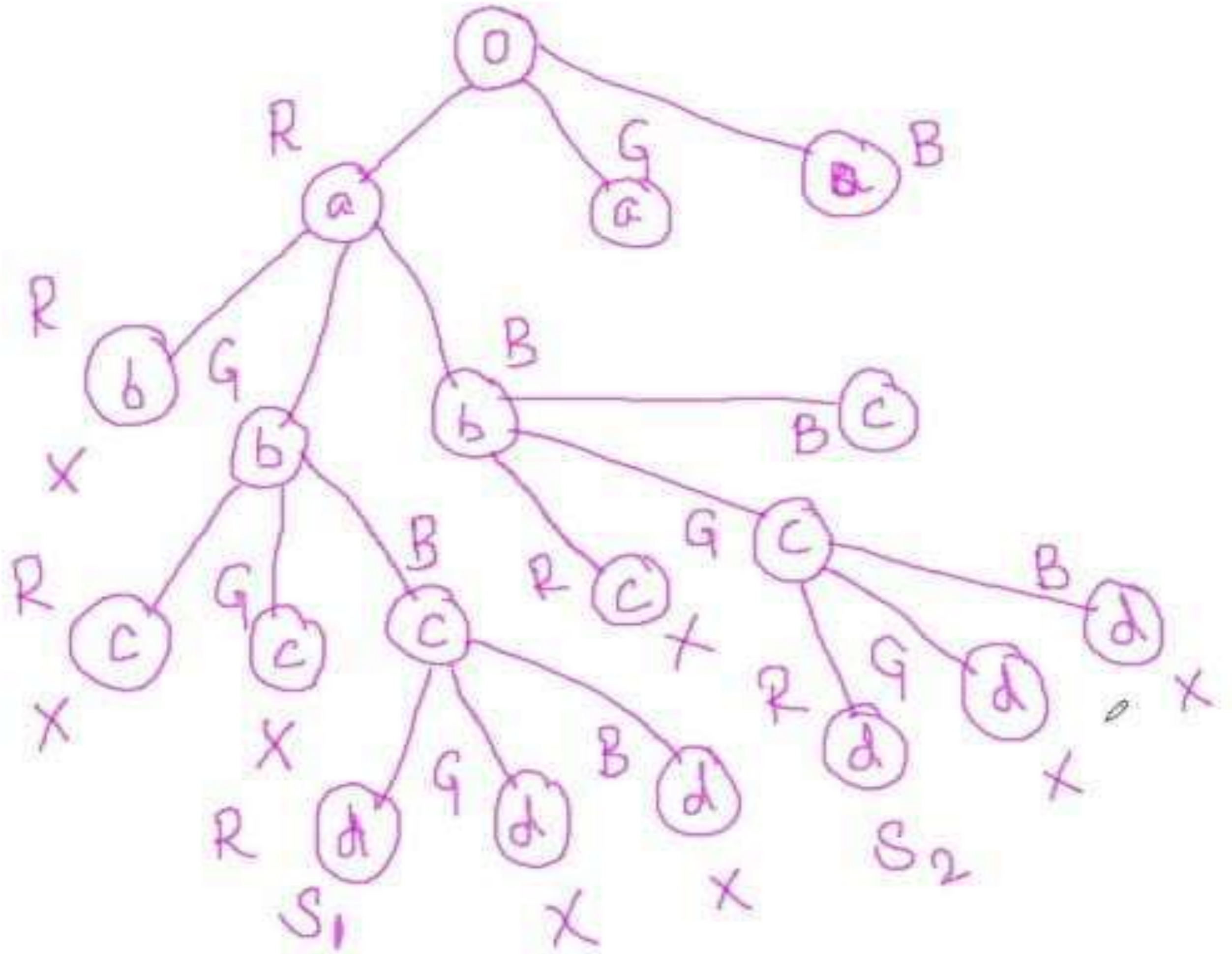
BACKTRACKING - GRAPH COLOURING PROBLEM



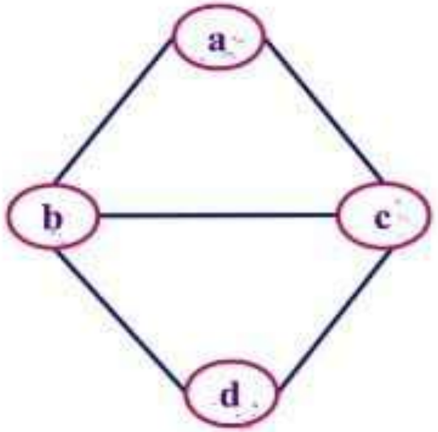
M=3

R,G,B

State Space Tree



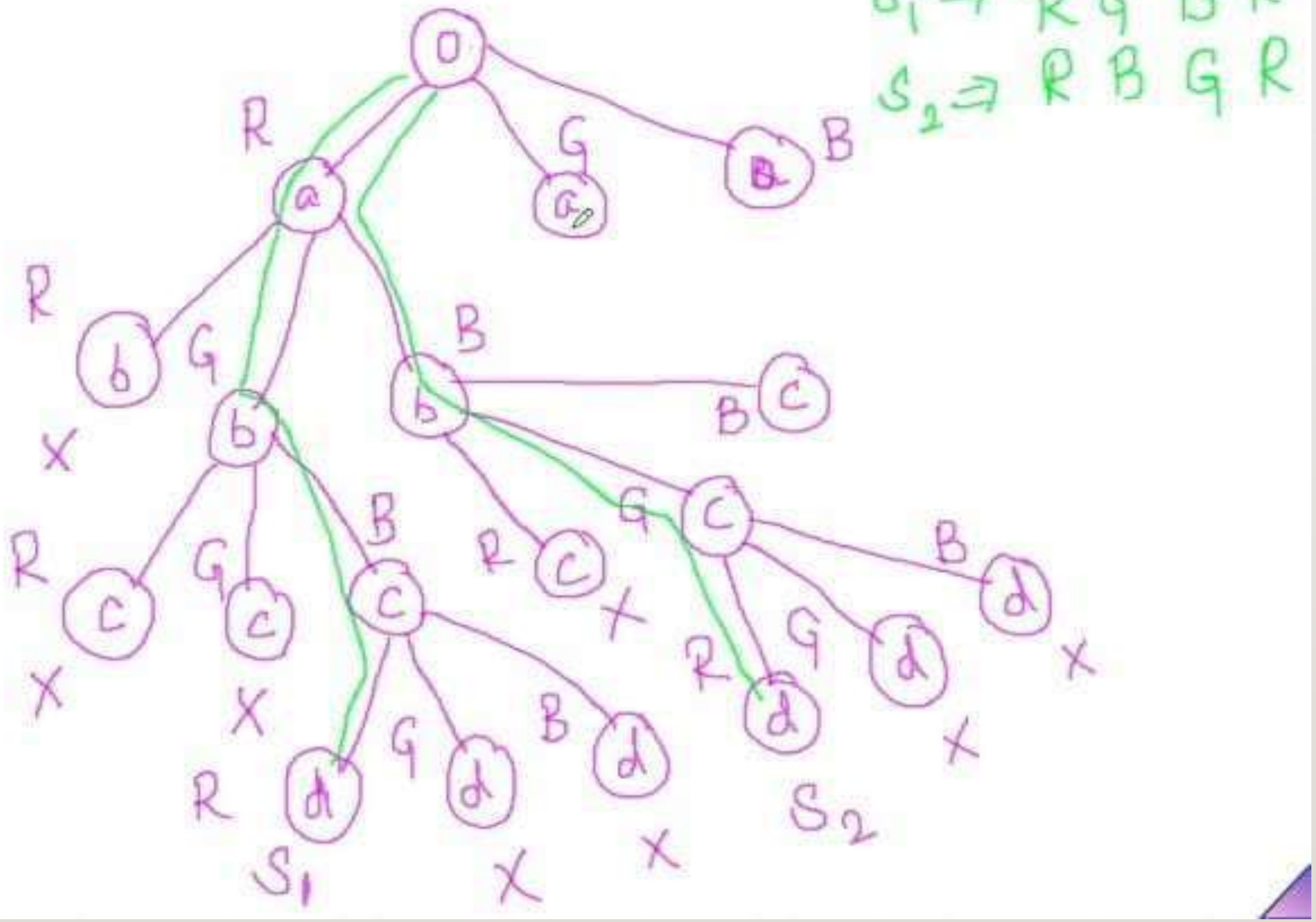
BACKTRACKING - GRAPH COLOURING PROBLEM



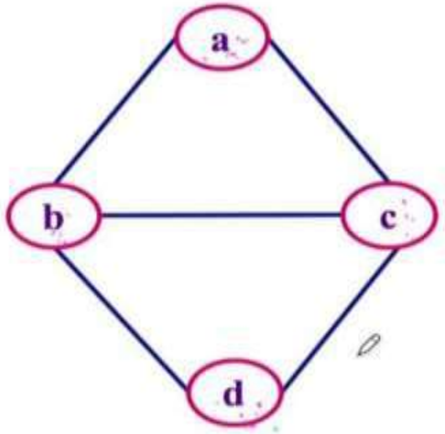
M=3

R,G,B

State Space Tree



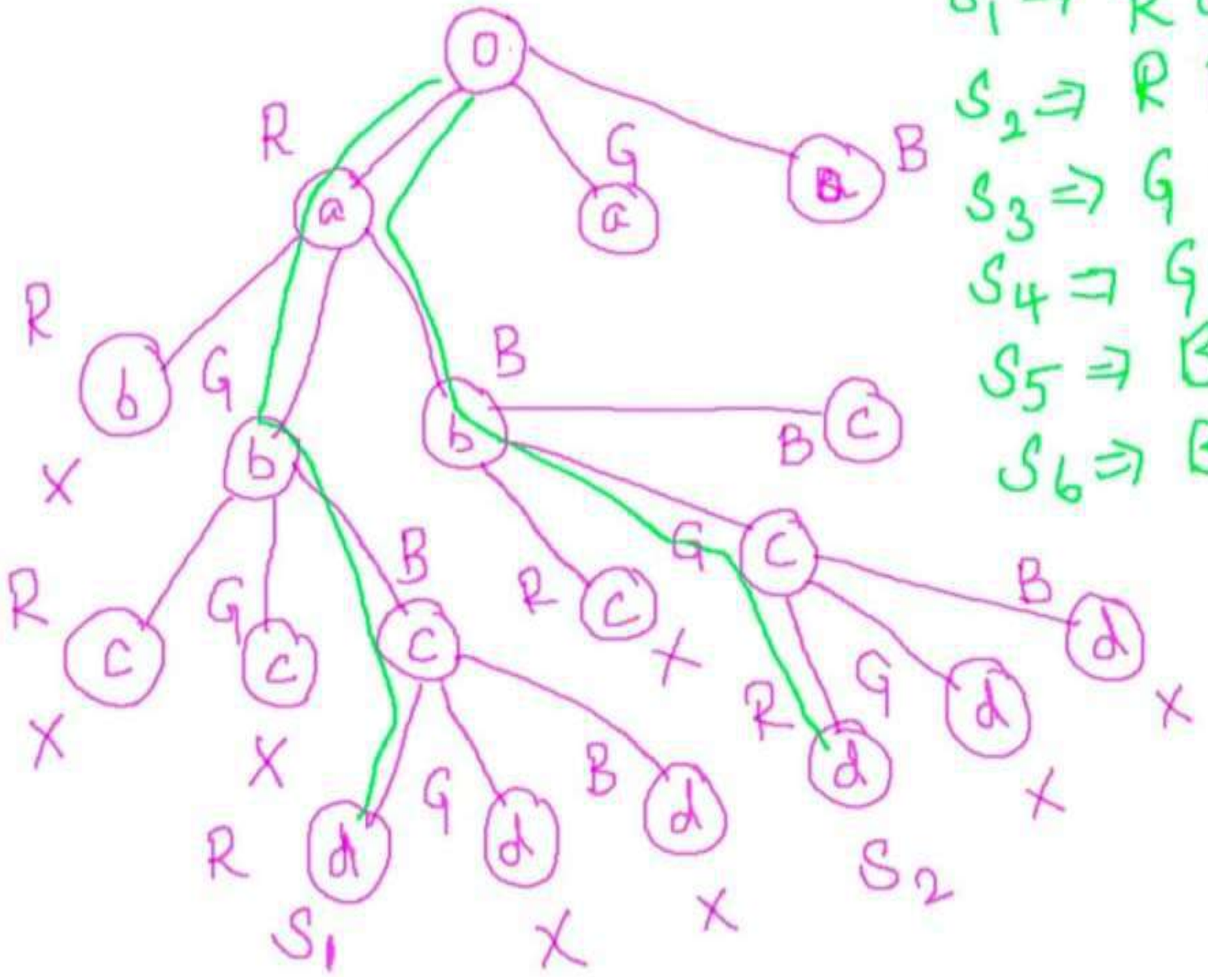
BACKTRACKING - GRAPH COLOURING PROBLEM



M=3

R,G,B

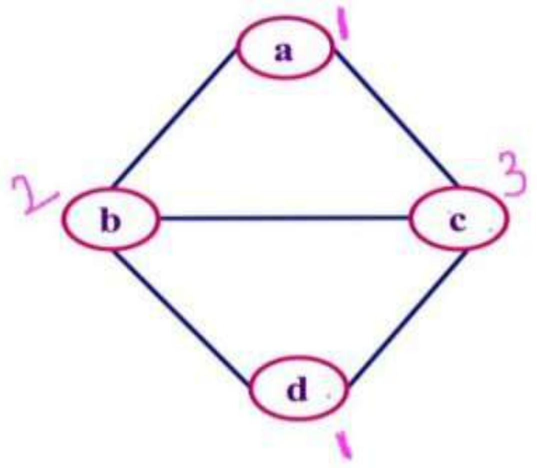
State Space Tree



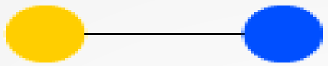
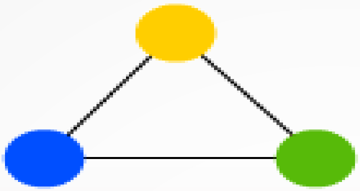
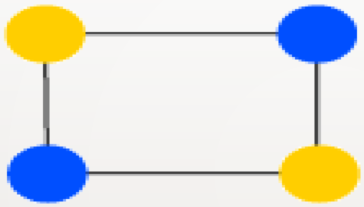
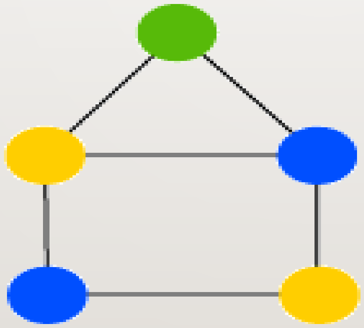
	a	b	c	d
$S_1 \Rightarrow$	R	G	B	R
$S_2 \Rightarrow$	R	B	G	R
$S_3 \Rightarrow$	G	R	B	G
$S_4 \Rightarrow$	G	B	R	G
$S_5 \Rightarrow$	B	G	R	B
$S_6 \Rightarrow$	B	R	G	B

BACKTRACKING - GRAPH COLOURING PROBLEM

What is the chromatic number for this given graph



Minimum No. of Colours Required ⇒ 3

Graphs	Chromatic Number
	2
	3
	2
	3

Chromatic number of graphs having different number of nodes

Algorithm mColoring(k)

```
// the graph is represented by its Boolean adjacency matrix G[1:n,1:n]  .All assignments
//of 1,2,……….,m to the vertices of the graph such that adjacent vertices are assigned
//distinct integers are printed. 'k' is the index of the next vertex to color.
{
repeat
{    // generate all legal assignment for X[k].
     Nextvalue(k);    // Assign to X[k] a legal color.
        If (X[k]=0) then return;          // No new color possible.
        If (k=n) then // Almost 'm' colors have been used to color the 'n' vertices
              Write(x[1:n]);
        Else mcoloring(k+1);
}until(false);
}
```

```
Algorithm NextValue(k)
// x[1], ..., x[k-1] have been assigned integer values in
// the range [1, m] such that adjacent vertices have distinct
// integers. A value for x[k] is determined in the range
// [0, m]. x[k] is assigned the next highest numbered color
// while maintaining distinctness from the adjacent vertices
// of vertex k. If no such color exists, then x[k] is 0.
{
    repeat
    {
        x[k] := (x[k] + 1) mod (m + 1); // Next highest color.
        if (x[k] = 0) then return; // All colors have been used.
        for j := 1 to n do
        {   // Check if this color is
            // distinct from adjacent colors.
            if ((G[k, j] ≠ 0) and (x[k] = x[j]))
            // If (k, j) is and edge and if adj.
            // vertices have the same color.
                then break;
        }
        if (j = n + 1) then return; // New color found
    } until (false); // Otherwise try to find another color.
}
```

TIME COMPLEXITY:

- An upper bound on the computing time of **mColoring** can be derived at by noticing that the number of internal nodes in the state space tree is:

$$\sum_{i=0}^{n-1} m^{i}$$

At each internal node, O(mn) time is spent by **NextValue** to determine the children corresponding to legal colorings.

$$\sum_{i=0}^{n-1} m^{i+1} n$$

Hence the total time is bounded by:

$$\sum_{i=0}^{n-1} m^{i} n$$

$$n(m^{n+1} - 2)/(m-1) = O(nm^{n})$$

SAMPLE QUESTIONS

- What is the Eight Queens problem? Explain its objective and constraints.
- Describe the rules and constraints that govern the placement of eight queens on an 8x8 chessboard.
- Explain the concept of backtracking
- What is goal state

A 4 – NODE GRAPH AND ALL POSSIBLE 3 COLORINGS