Design & Analysis of Algorithms
(Prof. Parminder Kaur Wadhwa)

Date ______
Page ______

Graph Coloring

Let G be a graph and m be a given positive integer.

We want to discover whether 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. This is called as m-colorability decision problem.

The m-colorability optimization problem asks for the smallest integer m for which the graph G can be colored. This integer is referred to as the CHROMATIC NUMBER of the graph.

Graph Coloring Problem

may be

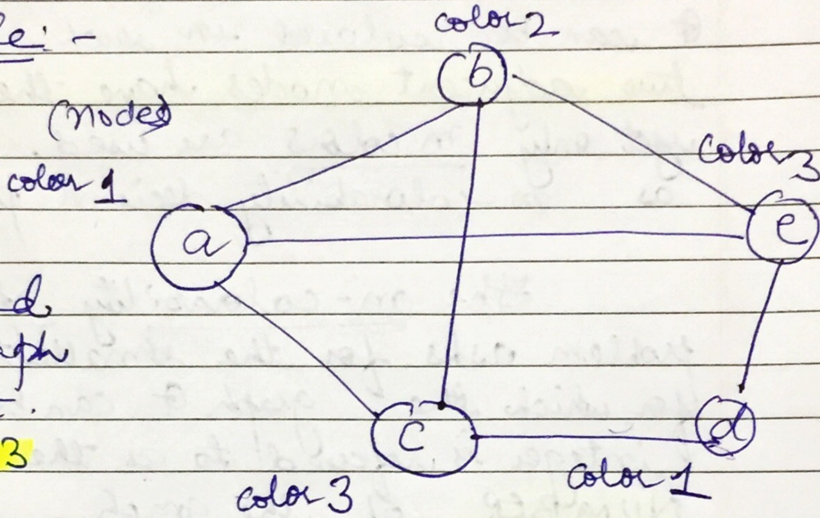
a decision problem	an optimization problem
(for a given number m, whether it is possible to color the graph such that no two adjacent nodes have same color)	(find the minimum value of m required to color the graph such that no two adjacent nodes have same color).
Yes or No?	

This has practical application in ==map coloring==. (No two regions which are adjacent to each other ~~th~~ should have the same color).

For example:-

Example 1:-

a) $n=5$ (nodes)

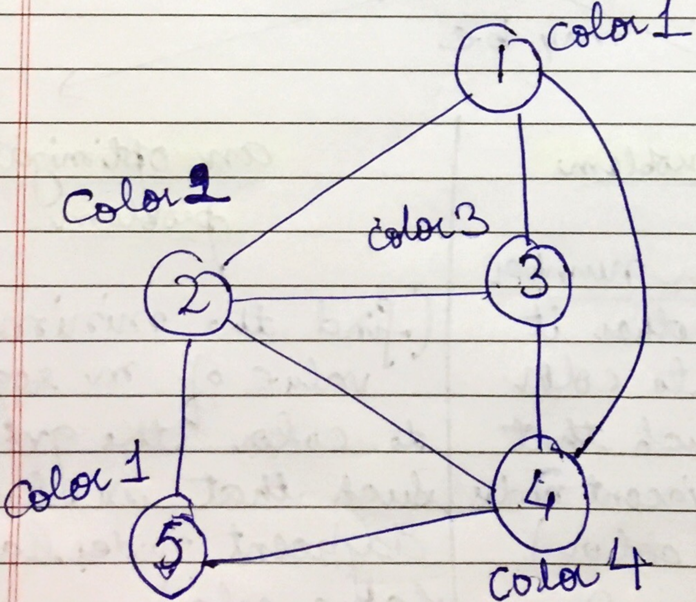


Three colors are required in this graph to color it.

$\therefore$ $m=3$

Example 2:-

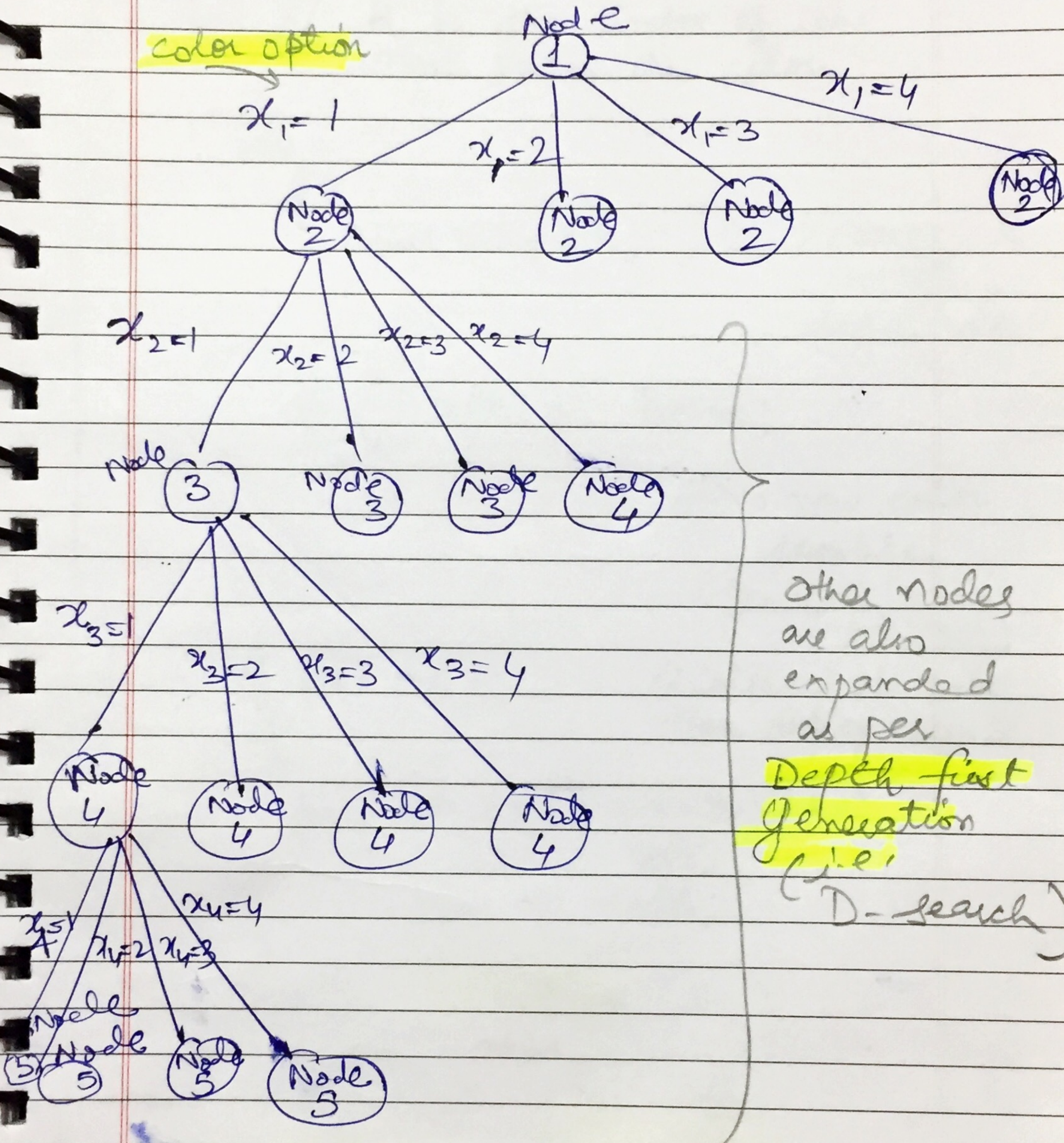
b) $n=5$ (no. of nodes or vertices = 5)



$m=4$ are required to color this graph

State Space Tree Graph Coloring Problem

(for $n=5$, $m=4$) (for example)



Date
Page

Graph Coloring

(A recursive backtracking algorithm)

```
void mColoring (int k)
{
  // k is the index of the
     next vertex to color

  do
  {
     NextValue (k);            // Assign to
                                  x[k] a
                                  legal color

     if (! x[k])   break;

                            // No new color
                               possible
                            (for decision
                                  problem)

    if (k == n)              // Colored all nodes
                                then display result
     {
       for (int i = 1; i <= n; i++)
              cout << x[i];
             cout << endl;
     }
       else  mColoring (k+1);
                      Recursive call to color
  } while (1);            next vertex
```

Worst case is $O(nm^n)$