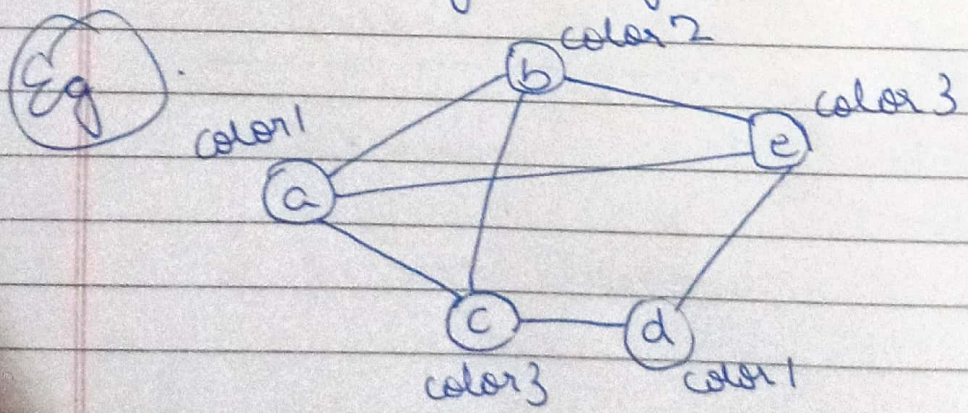
Jasleen kaur
1905345
D3I TA2

DAA Tut-5

Q5 What do you know about graph coloring problem? Explain how it can be solved using backtracking.

Ans: Let G be a graph and m be a given positive integer.
We want to discover whether the nodes of G can be coloured 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 ~~called~~ colored. This integer is referred to as the chromatic number of the graph.



Graph Coloring Problem

may be

a decision problem

(for a given number m, whether it is possible to color the graph such that no two adjacent nodes have same color)

an optimization problem

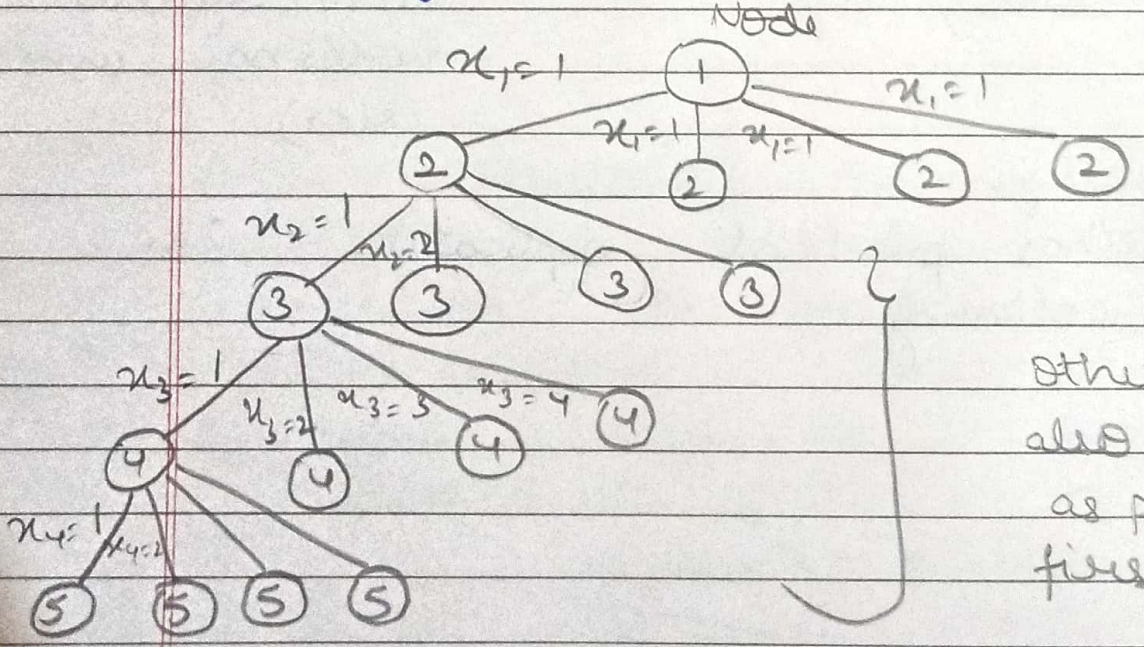
(find the minimum value of m required to color the graph such that no two adjacent nodes have same color)

This has practical application in map coloring.

Graph Coloring using backtracking

- By using the backtracking method, the main idea is to assign colors ~~and~~ one by one to different vertices right from the first vertex.
- Before color assignment, check if the adjacent vertices have same or different color by considering already assigned colors to the adjacent vertices.

Using depth first generation



Other nodes are also expanded as per depth first generation

Algorithm

```
void mcoloring (int k)
{
    // k is the index of the next vertex to color
    do
    {
        Next Value (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 next vertex
    } while (1);
}
```

Worst case time complexity $O(nm^n)$