Graph Coloring

DAA Lab 11

Introduction

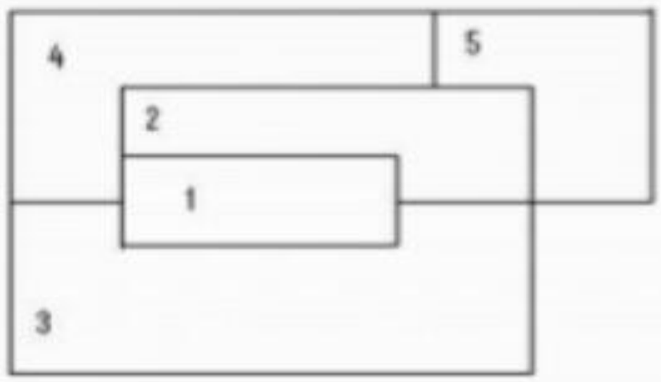
- **Graph coloring** is a special case of graph labeling. It is an assignment of labels traditionally called "colors" to elements of a graph subject to certain constraints. In its simplest form, it is a way of coloring the vertices of a graph such that no two adjacent vertices are of the same color; this is called a **vertex coloring**
- The other graph coloring problems like ***Edge Coloring*** (No vertex is incident to two edges of same color) and ***Face Coloring*** (Geographical Map Coloring) can be transformed into vertex coloring.
- **Chromatic Number:** The smallest number of colors needed to color a graph G is called its chromatic number. For example, the following can be colored minimum 3 colors.

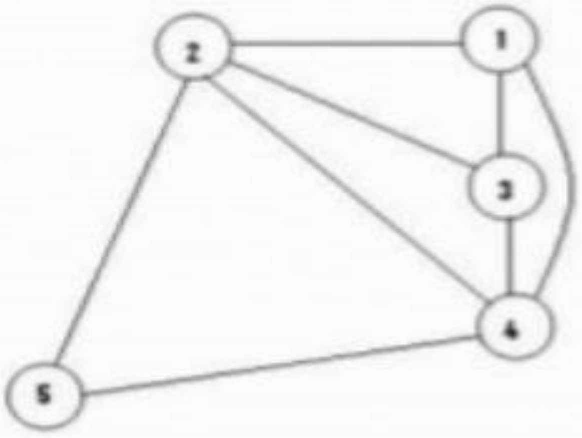
Coloring a map

Problem:

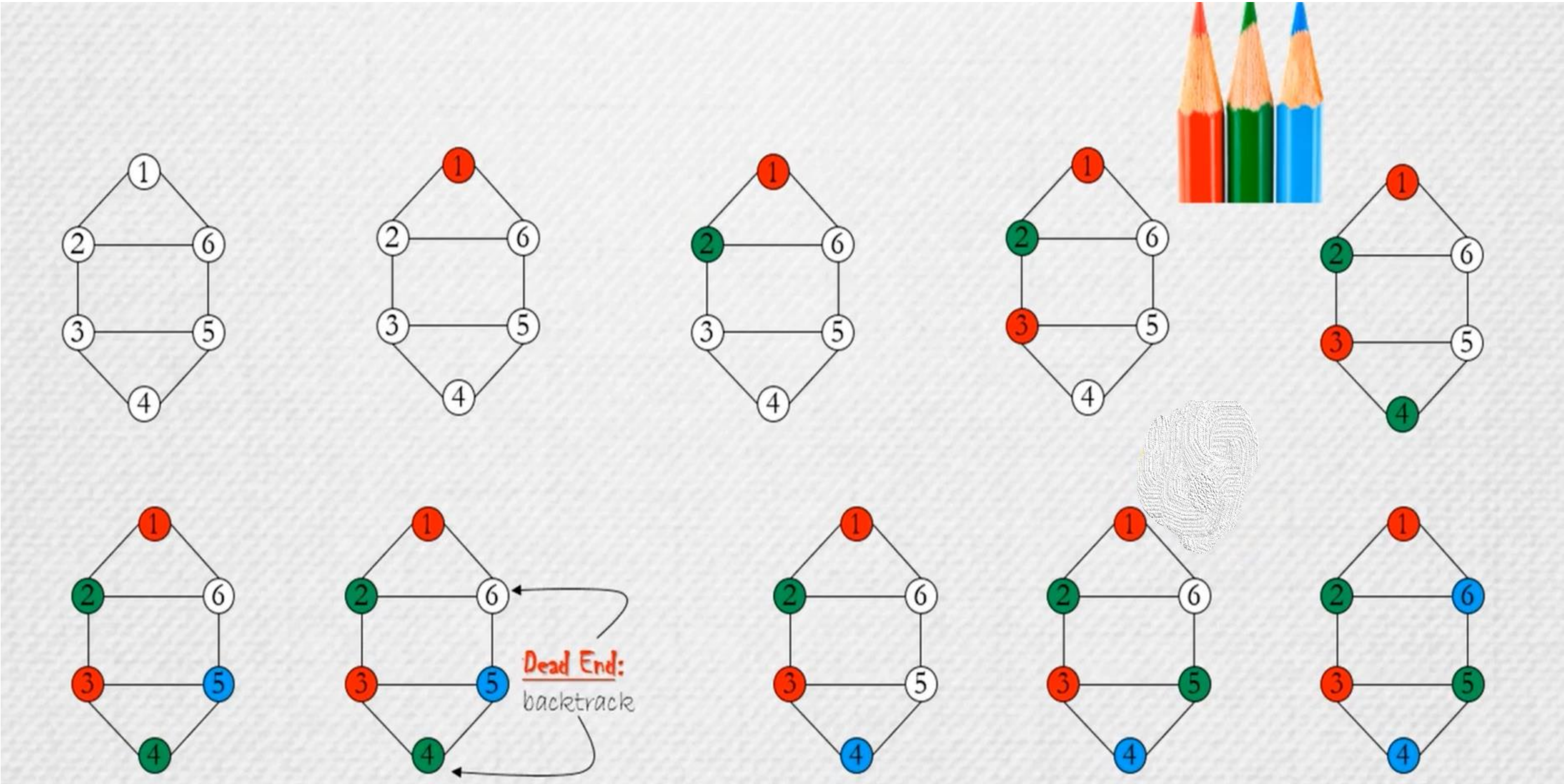
Let **G** be a graph and **m** be a given positive integer. We want to discover whether the nodes of **G** can be colored in such a way that no two adjacent node have the same color yet only **m** colors are used. This technique is broadly used in "map-coloring"; Four-color map is the main objective.

Consider the following map and it can be easily decomposed into the following planner graph beside it :



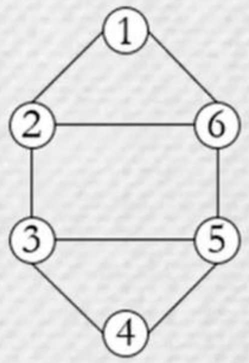


Graph coloring by backtracking with chromatic number 3

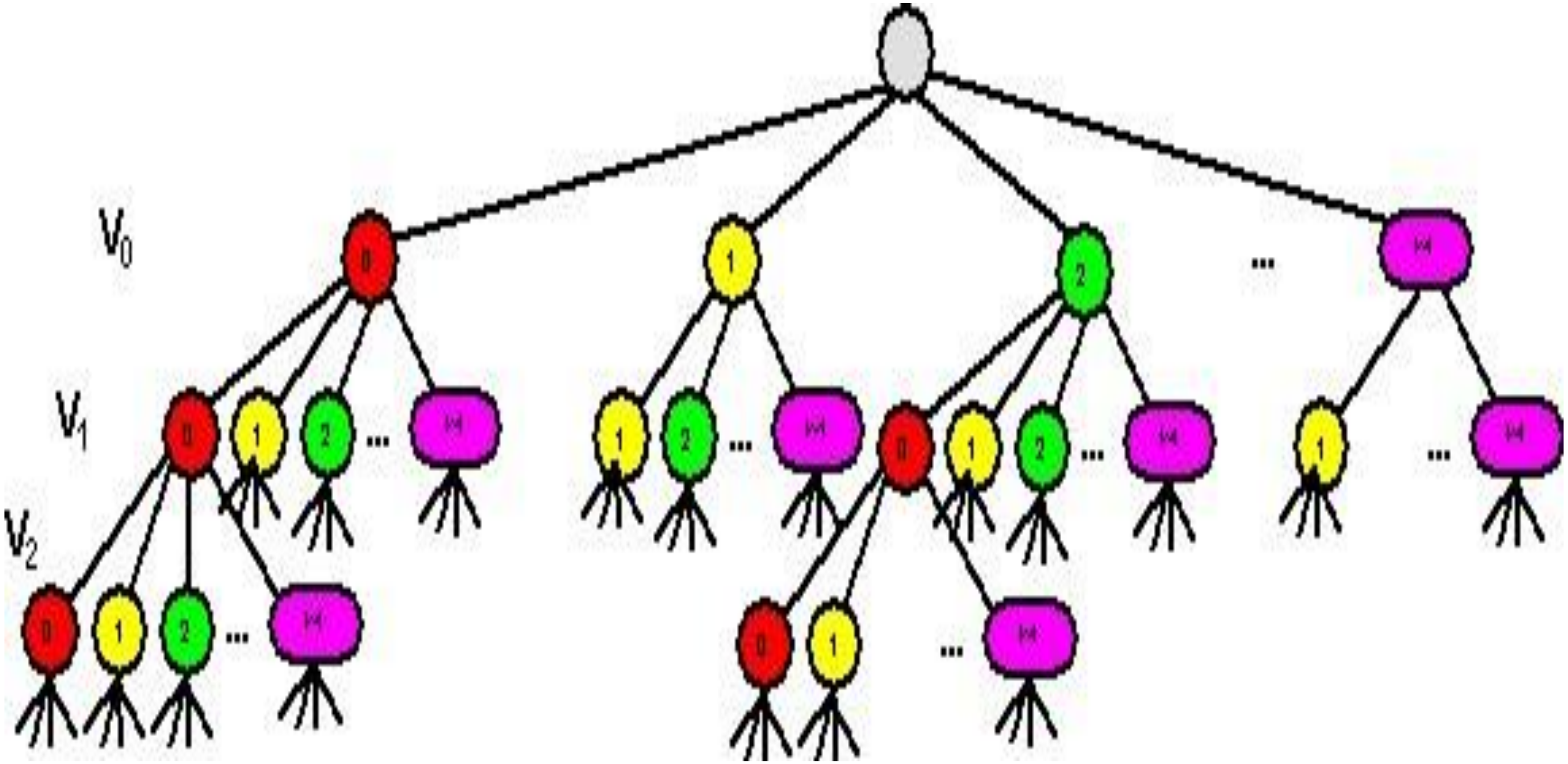


Designing the Graph coloring algorithm

- Vertices are numbered from 1 to n (n = 6 in our example)
- The adjacency matrix of the graph is represented by the two-dimensional array G[1 … n][1 … n]
- Colors are numbered from 1 to m (m = 3 in our example)
- e.g. color 1 means red, 2 means green and 3 means blue
- The solution is given by a one dimensional array color[1 … n]
- e.g. color[1] = 2 means vertex 1 is painted with color 2 (green)
- color[5] = 3 means vertex 5 is painted with color 3 (blue)
- Initially all values in the array color are set to 0.



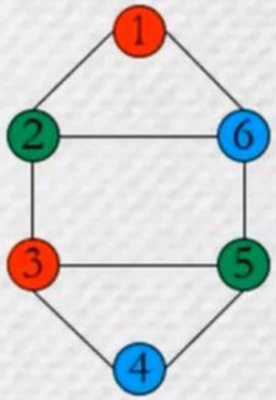
State space tree solution

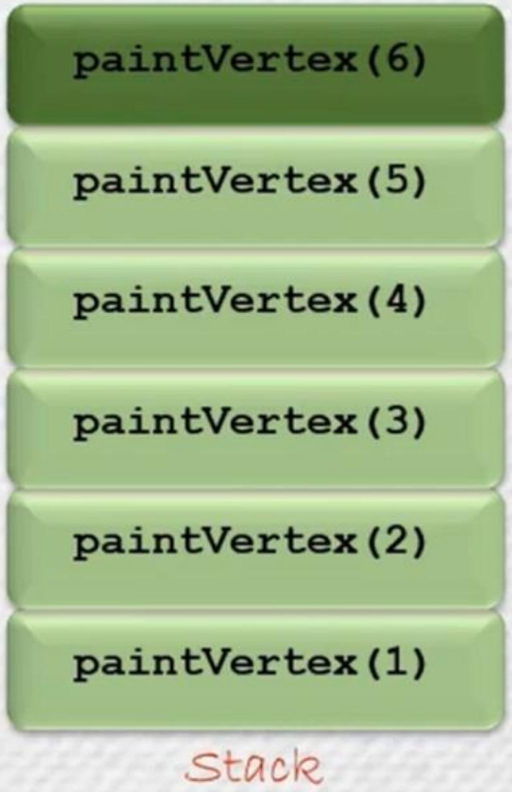


```
Algorithm chooseColorForVertex(k)
    do
        color[k] = (color[k] + 1) mod (m + 1)
        if (color[k] == 0) return
        for (j = 1 to n)
            if (G[k][j] ≠ 0  AND color[k] == color[j])
                break
            if (j == n + 1) return
    while (true)
```

```
Algorithm paintVertex(k)
    do
        chooseColorForVertex(k)
        if (color[k] == 0) return
        if (k == n) displayGraph()
        else paintVertex(k + 1)
    while (true)
```

	1	2	3	4	5	6
color	1	2	1	3	2	3





Prelab Questions:

1. Illustrate the backtracking technique to m-coloring graph.

2. Suppose want to schedule some internal exams for CS courses with following course numbers:1007,3137,3157,3203,3261,4115,4118,4156. Suppose also that there are no students in common taking the following pairs of courses:1007-3137,1007-3157,3137-3157 1007-3203,1007-3261,3137-3261,3203-3261,1007-4115,3137-4115,3203-4115,3261-4115,1007-4118,3137-4118,1007-4156,3137-4156,3157-4156 How many exam slots are necessary to schedule exams?

3. Describe graph coloring problem with greedy approach

4. Define chromatic number

5. Draw the state space tree for the following graph