Department of
Computer Science and Engineering

Title: Implement Graph Coloring Algorithm

Artificial Intelligence Lab
CSE 404

Green University of Bangladesh

1 Objective(s)

- To understand how to represent a graph using adjacency list.
- To understand how Graph Coloring Algorithm works.

2 Problem analysis

Graph coloring problem is to assign colors to certain elements of a graph subject to certain constraints.
Vertex coloring is the most common graph coloring problem. The problem is, given m colors, find a way of coloring the vertices of a graph such that no two adjacent vertices are colored using same color. 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 2 colors.

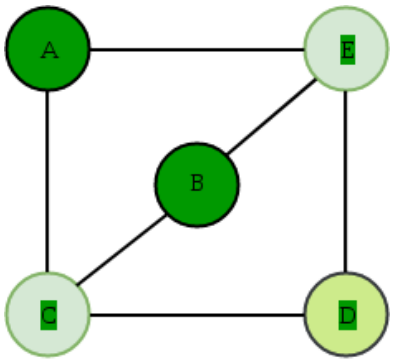


Figure 1: A simple graph

Adjacency List:
Vertices are labelled (or re-labelled) from 0 to V(G)- 1. Corresponding to each vertex is a list (either an array or linked list) of its neighbours. Table: 1 represents the adjacency list of figure1.

A to	D, S
B to	E, S
C to	F, S
D to	A, G
E to	B, G
F to	C, G
G to	D, E, F
S to	A, B, C

Table:1

3 Algorithm

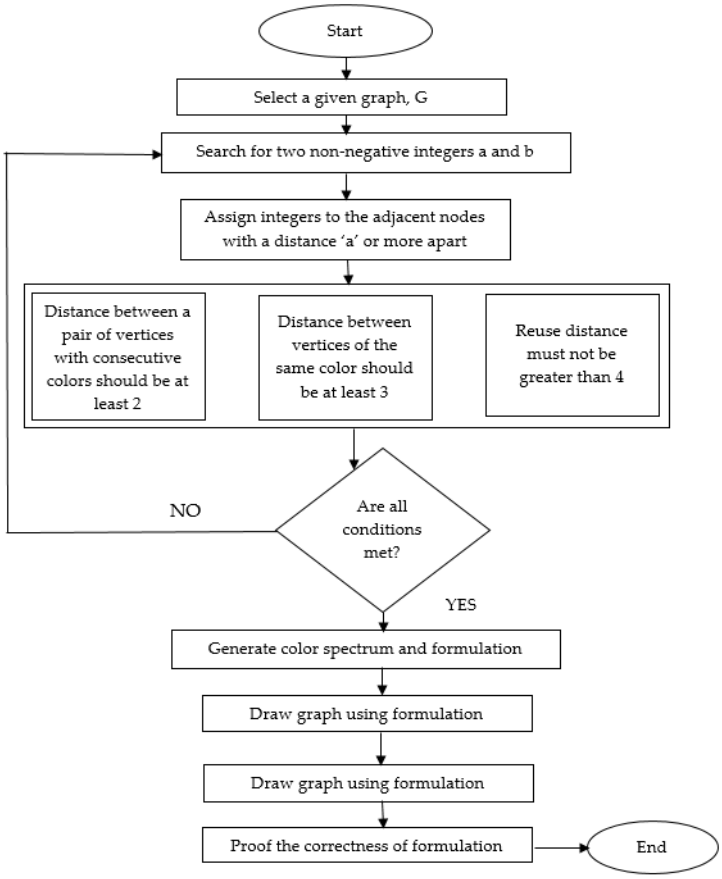
Algorithm 1: Graph Coloring Algorithm

```
/* Graph Coloring Algorithm                                          */
```

Input:
$G = (V, E)$ An undirected simple graph
Output:
A vertex coloring for graph G

begin
  Let $\{v_1, v_2, \ldots, v_n\}$ be a sequence of the vertices in V.
  for $v := v_1$ **to** v_n
  begin
    Assign vertex v the smallest possible color such that
    no conflict exists between v and its colored neighbors.
  end
end

4 Flowchart



5 Implementation in Java

```
1 package com.GraphColoring;
2
```

```java
import java.util.Scanner;

/** Class GraphColoring **/
public class GraphColoring
{
    private int V, numOfColors;
    private int[] color;
    private int[][] graph;

    /** Function to assign color **/
    public void graphColor(int[][] g, int noc)
    {
        V = g.length;
        numOfColors = noc;
        color = new int[V];
        graph = g;

        try
        {
            solve(0);
            System.out.println("No solution");
        }
        catch (Exception e)
        {
            System.out.println("\nSolution exists ");
            display();
        }
    }
    /** function to assign colors recursively **/
    public void solve(int v) throws Exception
    {
        /** base case - solution found **/
        if (v == V)
            throw new Exception("Solution found");
        /** try all colours **/
        for (int c = 1; c <= numOfColors; c++)
        {
            if (isPossible(v, c))
            {
                /** assign and proceed with next vertex **/
                color[v] = c;
                solve(v + 1);

                color[v] = 0;
            }
        }
    }
    /** function to check if it is valid to allot that color to vertex **/
    public boolean isPossible(int v, int c)
    {
        for (int i = 0; i < V; i++)
            if (graph[v][i] == 1 && c == color[i])
                return false;
        return true;
    }
    /** display solution **/
    public void display()
    {
```

```
        String [] textColor = {"", "RED", "GREEN", "BLUE","YELLOW", "ORANGE", "
            PINK",
                "BLACK", "BROWN", "WHITE", "PURPLE", "VIOLET"};
        System.out.print("\nColors : ");
        for (int i = 0; i < V; i++){
            System.out.print(textColor[color[i]] +" ");
        }

        System.out.println();
    }
    /** Main function **/
    public static void main (String[] args)
    {
        Scanner scan = new Scanner(System.in);
        System.out.println("Graph Coloring Algorithm Test\n");
        /** Make an object of GraphColoring class **/
        GraphColoring gc = new GraphColoring();

        /** Accept number of vertices **/
        System.out.println("Enter number of vertices\n");
        int V = scan.nextInt();

        /** get graph **/
        System.out.println("\nEnter matrix\n");
        int[][] graph = new int[V][V];
        for (int i = 0; i < V; i++)
            for (int j = 0; j < V; j++)
                graph[i][j] = scan.nextInt();

        System.out.println("\nEnter number of colors");
        int c = scan.nextInt();

        gc.graphColor(graph, c);

    }
}
```

6 Sample Input/Output (Compilation, Debugging & Testing)

Enter number of vertices

7

Enter matrix

0 1 1 0 0 0 0
1 0 1 1 0 0 0
1 1 0 1 1 1 0
0 1 1 0 1 0 0
0 0 1 1 0 1 0
0 0 1 0 1 0 0
0 0 0 0 0 0 0

Enter number of colors
3
Output:
Solution exists

Colors : RED GREEN BLUE RED GREEN RED RED

7 Lab Task (Please implement yourself and show the output to the instructor)

1. Use the following map to perform graph coloring algorithm
-Hints: convert it into Adjacency List

8 Lab Exercise (Submit as a report)

- Write a program to perform graph coloring algorithm which take input as text file from computer.

9 Policy

Copying from internet, classmate, seniors, or from any other source is strongly prohibited. 100% marks will be *deducted* if any such copying is detected.