To find out the minimum face-coloring of a given graph

Yash Sharma, Shubham Vashishtha, Aditya Kumawat

IT Department,Indian Institute of Information Technology,Allahabad
Deoghat,Jhalwa-211015 India
icm2014001@iiita.ac.in
ihm2014007@iiita.ac.in
ihm2014006@iiita.ac.in

Abstract—*This report proposes approach to find out the minimum face coloring of a given graph.*

Keywords — *face-coloring, graph, chromaticity, heuristic, greedy*

Motivation

In computer science, the face of a graph is a set of vertices that form a cycle. The face coloring problem is very important because many times we need to find minimum number of colors required to color the objects so that no two adjacent faces of the object are of same color. Coloring two adjacent faces differently is helpful in visually differentiating between them. Minimum number of colors is imperative for keeping the coloring as inexpensive as possible.

Problem Description

For a given graph and list of its faces, our aim is to calculate the minimum number of colors required to color all the faces of the graph such that no two adjacent faces are of same color.

Approach

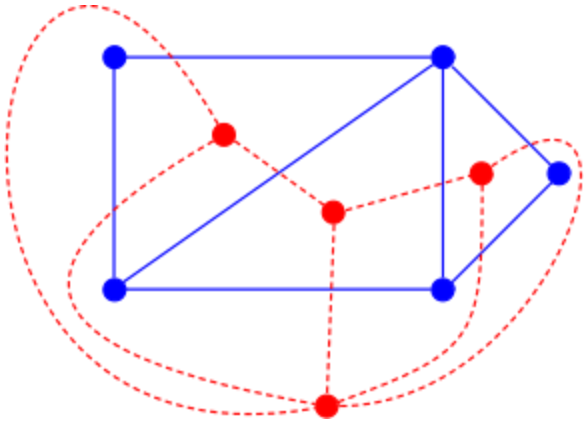
The problem of face coloring cannot be solved if only the adjacency or incidence matrix of the graph is given. This is because a graph can exist in with many possible faces in a given plane. Therefore we require the exact planar embedding of the graph to solve the face coloring problem. So, the input will be the set of faces of the graph.

We propose following algorithm face-coloring of the graph:

Proposed Approach : The solution to the problem of face coloring can be divided into two parts:

1. Converting a given graph to its dual graph.
2. Solving vertex coloring problem for the dual graph.

We propose one approach for solving the first part of the problem and two approaches for solving the second part. For the first part the input given is the set of faces of the graph. For every face of the given graph there is a corresponding vertex in the dual graph. Now, for every pair of faces having at least one edge in common, there is an edge in the dual graph between the vertices corresponding to the faces. We ignore the parallel edges in the given graph because they are not relevant to our problem.



The red graph is the dual graph of the blue graph, and vice versa [1]

For the next part of the problem, the input is the adjacency matrix of the dual graph. Now, we use two approaches to solve the vertex coloring problem. The first approach is Greedy and the other is heuristic approach given by Welsh-Powell which is an improvement over the first approach. The problem of vertex coloring is NP complete so these algorithms are not guaranteed to give optimal solution.

Part 1

We are given the list of faces in a graph. For every edge in every face, we will first create a data structure which represents the edge mapped with the faces, i.e. which edge is present in which face.
After creating the data structure defined above, take every pair of vertices from the data structure and create edge between corresponding vertices.

Algorithm : To create a dual graph from the set of faces of the given graph

Input : Array of faces of the graph
Output : Adjacency matrix of the corresponding dual graph

Pseudo Code :

```
vvi a(n);
vvi b(n, vi(n, 0));
map<ii, set<int> > edgeFaces;

For i=0 to n-1
       int m = a[i].size();
       for j=0 to m-1
              edgeFaces[ii(a[i][j],
a[i][(j+1)%m])].push_back(i);

For it=edgeFaces.begin() to =edgeFaces.end()
       set<int> ver = it->S;
       For i=ver.begin() to ver.end()
              for(auto j=i+1;j!=ver.end();j++)
                     b[*i][*j] = 1;
                     b[*j][*i] = 1;
```

Time Complexity Analysis: O(nV)

Part 2

PROBLEM - TO COLOR VERTICES OF A GRAPH SUCH THAT NO TWO ADJACENT VERTICES HAVE THE SAME COLOR

ALGORITHM 1: GREEDY APPROACH

INPUT : ADJACENCY MATRIX OF THE GIVEN GRAPH

OUTPUT : MINIMUM COLORS NEEDED TO COLOR A GRAPH

Pseudo Code :

```
greedyColoring():
  int result[V];
  result[0] = 0;
  for u = 1 to V-1
    result[u] = -1;
  bool available[V];
  for cr = 0 to V-1
    available[cr] = false;
  for u = 1 to V-1
    list<int>::iterator i;
    for i = adj[u].begin() to adj[u].end()
      if (result[*i] != -1)
        available[result[*i]] = true;
    for cr = 0 to V-1
      if (available[cr] == false)
        break;
    result[u] = cr;
    for i = adj[u].begin() to adj[u].end()
      if (result[*i] != -1)
        available[result[*i]] = false;
  return result;
```

Time Complexity Analysis: O(N^2 + E)

ALGORITHM 2: USING WELSH-POWELL HEURISTIC

INPUT : ADJACENCY MATRIX OF THE GIVEN GRAPH

OUTPUT : MINIMUM COLORS NEEDED TO COLOR A GRAPH

Pseudo Code :

```
1. Find the valencies of each vertex.
2. List the vertices in order of descending
degree (For conflict resolution if two or more
vertices have same degree then we can
select either one).
3. Color the first vertex in the list (the vertex
with the highest valence) with color 1.
4. Go down the list and color every vertex not
connected to the colored vertices above the same
color. Then cross out all colored vertices in the
list.
5. Repeat the process on the uncolored vertices
with a new color — always working in descending
order of degree until all the vertices have been
colored.
```

Time Complexity Analysis: For a complete graph, minimum N colors are required and for each color we will iterate the list of Size N so time complexity will be O(N^2).

1. CONCLUSION

The problem of vertex-coloring is NP complete. Therefore, the approaches used in solving the face-coloring problem are not guaranteed to give the optimal solution. For part two of the problem, heuristic is an improvement over greedy approach as it tries to avoid conflicts by picking up first vertices having high valency.

6. REFERENCES

1. https://en.wikipedia.org/wiki/Dual_graph
2. *http://mrsleblancsmath.pbworks.com/w/file/fetch/46119304/vertex%20coloring%20algorithm.pdf*