K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Computer Engineering

Batch: A1 **Roll No.: 16010123012**

Experiment No.: 09

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Title: Study, Implementation, and Analysis of Graph Coloring Problem.

Objective: To learn the Backtracking strategy of problem solving for Graph Coloring Problem.

CO to be achieved:

CO2: Analyze and solve problems for divide and conquer strategy, greedy method, dynamic programming approach and backtracking and branch & bound policies.

Books/ Journals/ Websites referred:

1. **Ellis horowitz, Sarataj Sahni, S.Rajsekaran," Fundamentals of computer algorithm", University Press**
2. **T.H.Cormen ,C.E.Leiserson,R.L.Rivest and C.Stein," Introduction to algortihtms",2nd Edition ,MIT press/McGraw Hill,2001**

Pre Lab/ Prior Concepts:
Data structures, Concepts of algorithm analysis

Historical Profile: The Graph Coloring Problem is a classical problem in graph theory and combinatorics with origins rooted in practical applications and mathematical curiosity. It has a rich history, spanning over two centuries, and remains a vibrant area of research due to its theoretical significance and real-world applications.

Origins and Early History: Map Coloring and the Four Color Theorem (1852):The problem of graph coloring originated from an attempt to color regions on maps so that no two adjacent regions share the same color. In 1852, Francis Guthrie, a British mathematician, conjectured the Four Color Theorem, stating that four colors are sufficient to color any map in a plane.

Graph Representation of Maps: In 1879, Arthur Cayley formulated the map coloring problem in terms of graph theory, representing regions of a map as vertices and adjacency as edges.

New Concepts to be learned:
Application of algorithmic design strategy to any problem, Backtracking method of problem solving Vs other methods of problem solving problem sum of subset and its applications.

Algorithm:

Backtracking Algorithm: The backtracking approach finds the optimal solution by trying all possible color assignments.

Steps:

1. Try assigning each vertex a color from 1 to k, where k is the number of colors.
2. Backtrack if an assignment leads to a conflict (two adjacent vertices having the same color).
3. Continue until all vertices are colored or all possibilities are exhausted.

Implementation (Code):

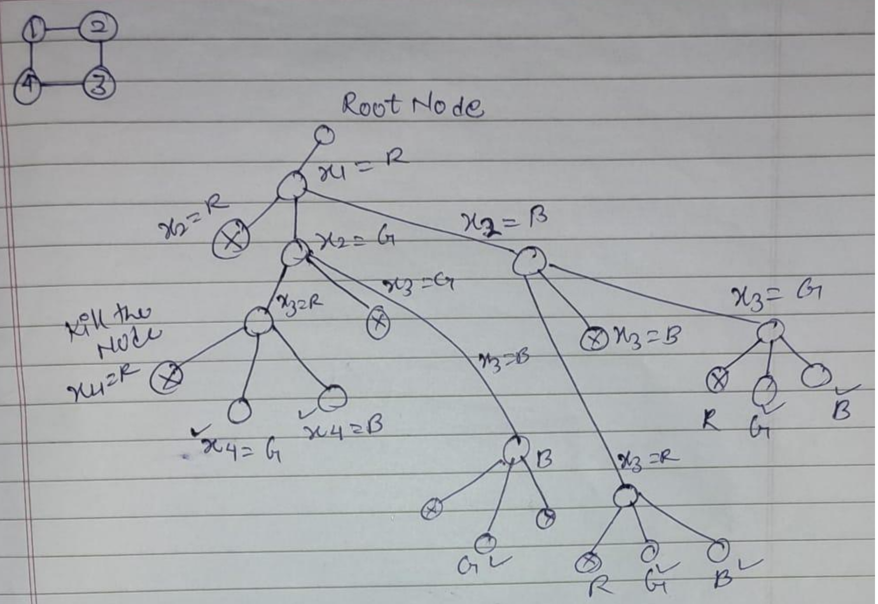
```
#include <bits/stdc++.h>
using namespace std;

bool isSafe(int v, const vector<vector<int>> &graph, const vector<int>
&color, int c)
{
  for (int neighbor : graph[v])
  {
    if (color[neighbor] == c)
    {
      return false;
    }
  }
  return true;
}

bool graphColoringUtil(const vector<vector<int>> &graph, int m,
vector<int> &color, int v)
{
  if (v == graph.size())
  {
    return true;
  }

  for (int c = 1; c <= m; c++)
  {
    if (isSafe(v, graph, color, c))
    {
      color[v] = c;
      if (graphColoringUtil(graph, m, color, v + 1))
      {
        return true;
```

```
      }
      color[v] = 0;
    }
  }
  return false;
}

void graphColoring(const vector<vector<int>> &graph, int m)
{
  vector<int> color(graph.size(), 0);

  if (!graphColoringUtil(graph, m, color, 0))
  {
    cout << "No valid coloring exists with " << m << " colors.\n";
    return;
  }

  cout << "Valid coloring found with " << m << " colors:\n";
  for (int i = 0; i < graph.size(); i++)
  {
    cout << "Vertex " << i << " -> Color " << color[i] << endl;
  }
}

int main()
{
  int V, E, m;

  cin >> V >> E;

  vector<vector<int>> graph(V);

  for (int i = 0; i < E; i++)
  {
    int u, v;
    cin >> u >> v;
    graph[u].push_back(v);
    graph[v].push_back(u);
  }

  cin >> m;

  graphColoring(graph, m);
  return 0;
}
```

Output:

```
4 5
0 1
0 2
0 3
1 2
2 3
3
Valid coloring found with 3 colors:
Vertex 0 -> Color 1
Vertex 1 -> Color 2
Vertex 2 -> Color 3
Vertex 3 -> Color 2
```

Example sum of subset Problem along with state space tree:



• DFS
Backtracking
go back
$x_1 = R$ has 6 solutions
similarly $x_1 = G$ has 6 solutions
$x_1 = B$ has 6 solutions
Total solutions = 6 + 6 + 6 = 18

1) RGRG
2) RGRB
3) RGBG
4) RBRG
5) RBRB
6) RBGB
7) GBGB
8) GBGR
9) GRGB
10) GBRB
11) GRGR
12) GBBR
13) BRBR
14) BRBG
15) BRGR
16) BGBR
17) BGBG
18) BGRG

Analysis of Backtracking solution for:
Time: Exponential (O(m^V)), V is vertex and m is colors
Space: Linear (O(V))

Conclusion:
I have successfully completed the experiment on the Graph Coloring Problem using the Backtracking approach. Through this experiment, I gained a deeper understanding of how to apply the backtracking algorithm to solve constraint satisfaction problems efficiently. I also explored the implementation of graph coloring in C++ and analyzed the time and space complexity involved.