Batch: B1 Roll No.: 1711072

Experiment No. 9

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

Signature of the Staff In-charge with date

Title: Implementation of Backtracking Algorithm

Objective: To learn the Backtracking strategy of problem solving for Graph Colouring problem

CO to be achieved:

Sr. No	Objective
CO 1	Compare and demonstrate the efficiency of algorithms using asymptotic complexity notations.
CO 2	Analyze and solve problems for divide and conquer strategy, greedy method, dynamic programming approach and backtracking and branch & bound policies.
CO 3	Analyze and solve problems for different string matching algorithms.

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**
3. **http://www.math.utah.edu/~alfeld/queens/queens.html**
4. **http://www-isl.ece.arizona.edu/ece175/assignments275/assignment4a/Solving%208%20queen%20problem.pdf**
5. **http://www.slideshare.net/Tech_MX/8-queens-problem-using-back-tracking**
6. **http://www.mathcs.emory.edu/~cheung/Courses/170.2010/Syllabus/Backtracking/8queens.html**
7. **http://www.geeksforgeeks.org/backtracking-set-3-n-queen-problem/**
8. **http://www.hbmeyer.de/backtrack/achtdamen/eight.htm**

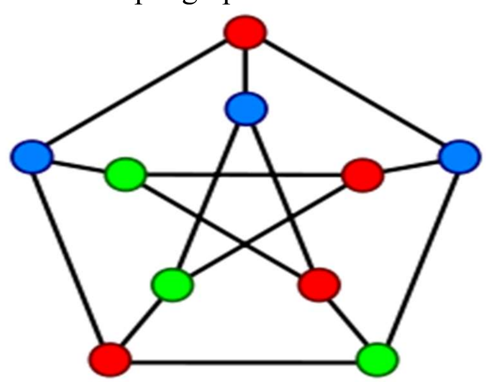
Pre Lab/ Prior Concepts:
Data structures, Concepts of algorithm analysis

Historical Profile:
Given an undirected graph and a number m, determine if the graph can be colored with at most m colors such that no two adjacent vertices of the graph are colored with same color. Here coloring of a graph means assignment of colors to all vertices.
Input:

1) A 2D array graph[V][V] where V is the number of vertices in graph and graph[V][V] is adjacency matrix representation of the graph.
Output:

An array color[V] that should have numbers from 1 to m. color[i] should represent the color assigned to the ith vertex. The code should also return false if the graph cannot be colored with m colors.

Following is an example graph can be colored with 3 colors.



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 graph colouring and its applications.

Algorithm Graph colouring Problem:-

```
Algorithm mColoring(k)
// This algorithm was formed using the recursive backtracking
// schema. The graph is represented by its boolean adjacency
// matrix G[1 : n, 1 : n]. All assignments of 1, 2, ..., m to the
// vertices of the graph such that adjacent vertices are
// assigned distinct integers are printed. k is the index
// of the next vertex to color.
{
    repeat
    {// Generate all legal assignments for x[k].
        NextValue(k); // Assign to x[k] a legal color.
        if (x[k] = 0) then return; // No new color possible
        if (k = n) then      // At most m colors have been
                             // used to color the n vertices.
            write (x[1 : n]);
        else mColoring(k + 1);
    } until (false);
}
```

Code:

```
states={0:"Maharashtra", 1:"Goa", 2:"Karnataka", 3:"Gujarat", 4:"Chattisgarh",
5:"Odisha", 6:"Bengal"}
graph={0:[1,2,3,4], 1:[0,2], 2:[0,1], 3:[0], 4:[0,5], 5:[4,6], 6:[5]}
colors=["Red", "Green", "Blue"]
col_graph,j={},1

def check(state, colour):
  global graph
  global col_graph
  for i in graph[state]:
    if i in col_graph and col_graph[i]==colour:
      return False
  return True

def assign(state, colour):
  global col_graph
  col_graph[state]=colour
```

```python
def solve(vertex):
  i=0
  global j
  if vertex==7:
    print('Solution ',j,':\n')
    for key, value in col_graph.items():
      print(states[key] + " : " + colors[value])
      if(states[key]=='Bengal'):
        print('\n')
    j=j+1
    return False
  for i in range(len(colors)):
    if check(vertex,i)==True:
      assign(vertex,i)
      if solve(vertex+1)==True:
        return True
      assign(vertex,0)
solve(0)
```

Screenshots:

```
Solution  24 :

Maharashtra : Blue
Goa : Green
Karnataka : Red
Gujarat : Red
Chattisgarh : Green
Odisha : Blue
Bengal : Green


Solution  25 :

Maharashtra : Blue
Goa : Green
Karnataka : Red
Gujarat : Green
Chattisgarh : Green
Odisha : Blue
Bengal : Red


Solution  26 :

Maharashtra : Blue
Goa : Green
Karnataka : Red
Gujarat : Green
Chattisgarh : Green
Odisha : Blue
Bengal : Green
```

Analysis of Backtracking solution for Graph Colouring Problem:

In the main logic of the program, we are comparing every vertex with its adjacent edge to check for colours. So, every such comparison yields a time complexity of

$$\mathbf{T(n) = O(V^2+E)}$$

where V is number of vertices and E is number of edges. We are displaying all possible solutions instead of showing one optimal solution with minimum colours. If optimal solution was to be shown, the time complexity would have been around $\mathbf{O(V^2)}$.

CONCLUSION: The graph colouring problem was solved using backtracking and all possible solutions for a given graph were generated.