Backtracking Assignment

Name : Shawn Louis **Batch : B** **Roll No : 31**

AOA Assignment-02 Date

(1) Explain Backtracking with N queen Problem.

Ans In backtracking method :

→ Desired solution is expressible as an n ~~tup~~ tuple $(x_1, x_2 \ldots x_n)$ where x_i is chosen from some finite set S_i.

→ The solution maximizes or minimizes or satisfies a certain criterion - function $C(x_1, x_2, \ldots x_n)$

The N-queen problem is to find an arrangement of N-queens on a chess-board of N×N such that no 2 queens can attack any other queens on the board.

The chess queens can attack in any direction as horizontal, vertical, diagonal way.

A binary matrix is used to display the positions of N-Queens, where no queens can attack other queens.

The case of 2 × 2 chess Board fails to give solution.

~~... we can get solution~~

Scanned with CamScanner

Algorithm:

→ isValid (board, row, col)

Begin

if there's queen at left of current col)

then return false.

if there's queen at left of upper diagonal

then return false

if there's queen at left of lower diagonal

then return false:

return true.

End

→ solveNQueen(board, col)

Begin

if all cols are filled, then

return true.

for each row of board, do

if isValid (board, i, col), then

set Queen at place (i, col) in board

if solveNQueen (board, col+1) = true

then,

return true.

otherwise remove queen from (i, col)

done

return false

End.

Scanned with CamScanner

(2) Write short Note on 8 queens problem. write an algorithm for the same.

Ans The 8 queens problem is the problem of placing 8 queens on an 8x8 chessboard such that none of them attack one another. ~~in the same row, colum~~

For this we use backtracking technique.

~~Alg~~ Hence the possible solution could be

Q							
		Q					
				Q			
						Q	
	Q						
			Q				
					Q		
							Q

	Q						
			Q				
					Q		
							Q
Q							
		Q					
				Q			
						Q	

Algorithm

```
Queen (n) {
for col ← 1 to n do
  { if (place (row, col)) then
     { board [row] ← col
     if (row = n) then
          print_board (n)
     else
          Queen (row+1, n)
     }
}
```

Algorithm Place (row, col)
i/p ⇒ row & col.
o/p ⇒ returns 0 for conflicting row and col position and 1 for no conflict.

```
for i ← 0 to row-1 do
  {
    if (board [i] = col) then
        return 0.
    else if (abs (board [i] - col) =
                  abs (i - row)) then
          return 0
  }
  return 1
```

(3) What is backtracking Approach? Explain how it is used in graph coloring?

Ans → Backtracking algorithms are used when we have set of choices and we don't know which choice will lead to a correct solution. The algorithm generates all partial candidates that may generate a complete solution.

→ The solution in backtracking is expressed as n-tuple $(x_1, x_2, \ldots x_n)$ where x_i is chosen from the finite set of choices S_i. Elements in solution tuple are chosen such that

Scanned with CamScanner

it maximize or minimize given criterion function $C(x_1, x_2 \ldots x_n)$

The idea in ~~the~~ 'graph coloring' is to assign colors one by one to different vertices, starting from vertex 0. Before assigning a color, we check for safety by considering already assigned colors to the adjacent vertices. If we find a color assignment which is safe, we mark the color assignment as part of solution. If we donot find color due to clashes then we backtrack and return false.

Given an undirected Graph we can also determine if graph can be colored with most m colors, using a Backtracking Algorithm.

The approach of this algorithm can be summarized as:

```
while there are untried configurations
{
    generate next configuration
    if no adjacent vertices are colored
       with same color
       {
          print this configuration;
       }
}
```

(4) Define chromatic number of graph. Explain Graph coloring algorithm.

Ans 'Graph coloring' problem is to assign colors to certain elements of a graph subject to certain constraints.

Chromatic Number:
The smallest number of colors needed to color a graph G is called its chromatic number. ~~For Ex~~, and it is denoted as X(G).

~~X(G) = 1 if and only if~~.

Algorithm:

```
mcoloring (k)
{
  repeat
  {
    Next value(k);
     If (x[k]=0) then return;
     If (k=n) then
      write (x[1:n]);
      Else mcoloring(k+1);
  }
  Until(False);
}
```

Scanned with CamScanner

This algorithm is 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 graph such that adjacent vertices are assigned distinct are printed. K is the index of the next vertex to color..

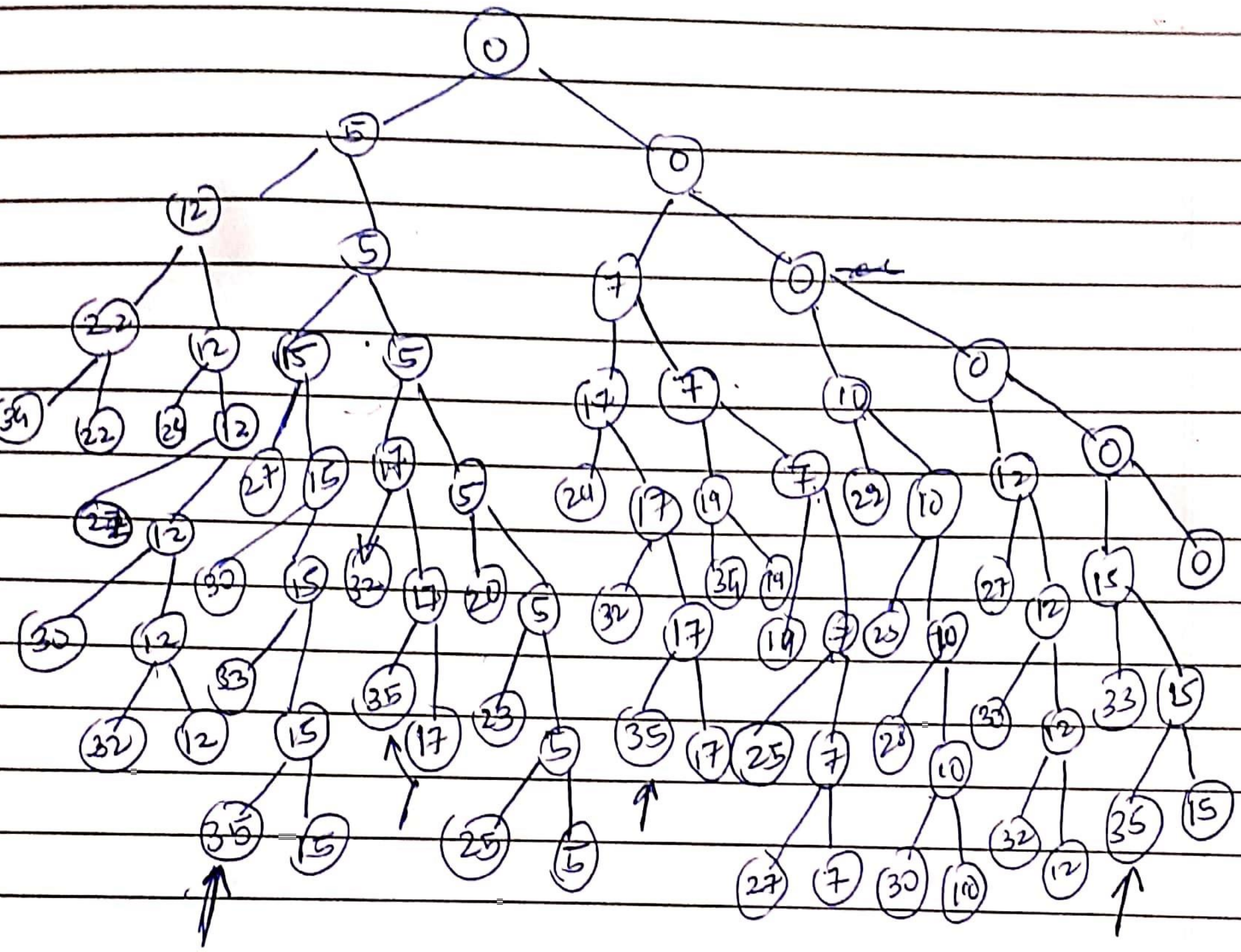
Total time required by this algorithm is $O(nm^n)$

(5) Solve sum of subset Problem and draw portion of state space tree.

(i) $W = \{5, 7, 10, 12, 15, 18, 20\}$ $M = 35$.
find all possible subsets of W that sum to M.

(ii) $N = 4$; $W = \{4, 5, 8, 9\}$ required sum = 9.

(6)



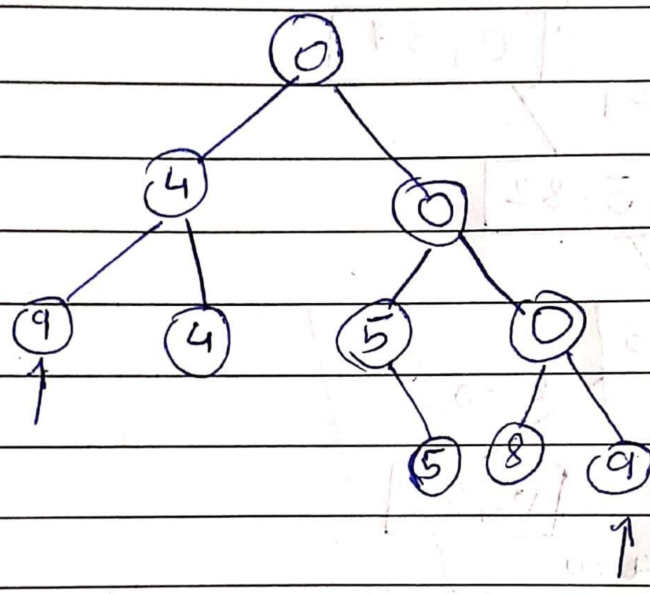
Hence we get four solutions.

$\{5, 10, 20\}$

$\{5, 12, 18\}$

$\{7, 10, 18\}$

$\{15, 20\}$



Hence we get two solution sets
{4, 5} , {9}

Scanned with CamScanner