UNIT-5

Backtracking: Backtracking is an algorithmic technique for solving problems recursively by trying to build a solution incrementally, one piece at a time, removing those solutions that fail to satisfy the constraints of the problem at any point of time.

Backtracking algorithm determines the solution by systematically searching the solution space for the given problem. Backtracking is a **depth-first search** with any bounding function. All solution using backtracking is needed to satisfy a complex set of constraints. The constraints may be explicit or implicit.

Explicit Constraint is ruled, which restrict each vector element to be chosen from the given set.

Implicit Constraint is ruled, which determine which each of the tuple in the solution space, actually satisfy the criterion function.

Algorithm:

```
if  r = n+1

   print Q[1…n]  //  we represents the positions of the queens using an array Q[1…n]

else

   for  j ← 1 to n

       legal ← true

       for i← to r --- 1

       if(Q[i] = j ) or (Q[i] = j + r-i) or (Q[i] = j – r + i) // Q[i] indicates which square in row I
contains a queen

      legal ← false

if legal

  Q[r] ← j

PLACE QUEENS(Q[1…n], r + 1)  // when PLACE QUEENS is called, the input parameters
r is the index of the first empty row and the prefix Q[1…r-1] contains the positions of the
first r-1 queens.
```

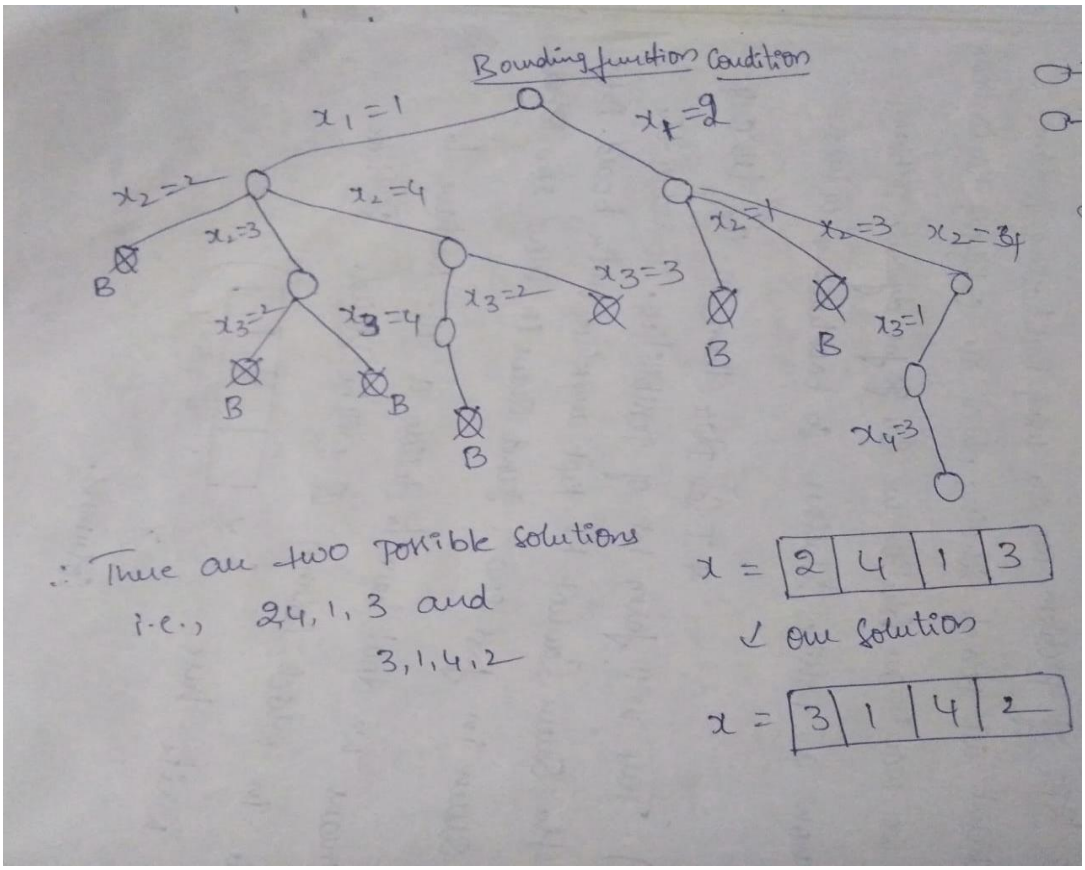
N Queens Problem: Let us understand what it means by "N-Queens Problem".

- The chessboard is given, of 4*4 means 16 cells. A standard chessboard will be 8*8 but for reducing a size of problem we are using 4*4. Here 4 queens are given (Q1Q2Q3Q4).
- So, in a game of chess, Queen will be there which will have its move like Horizontal, row or column, diagonal moves. We have to place that queen on this chessboard.
- So, all 4 queens we have to place such that, no two queens are under attack. When do you say they are under attack, if they are in same row or same column or they are in same diagonal.
- So, we have to avoid these and arrange themselves that, they are not under attack. It is possible to arrange? Yes, it is possible.
- We want all possible solutions which are satisfying this condition.

I can place them in 16c4 ways. This will form lot of possibilities. To reduce the size the queen cannot be kept anywhere on the board. But, the first queen in first row, second queen in the second row likewise.

It means we don't have to decide a cell. We have to decide in which column Q will be placed.

So, now checking all these conditions and prepares the solution in the form of "State space tree".



	1	2	3	4
1		Q1		
2				Q2
3	Q3			
4			Q4	

	1	2	3	4
1			Q1	
2	Q2			
3				Q3
4		Q4		

X =

2	4	1	3

3	1	4	2

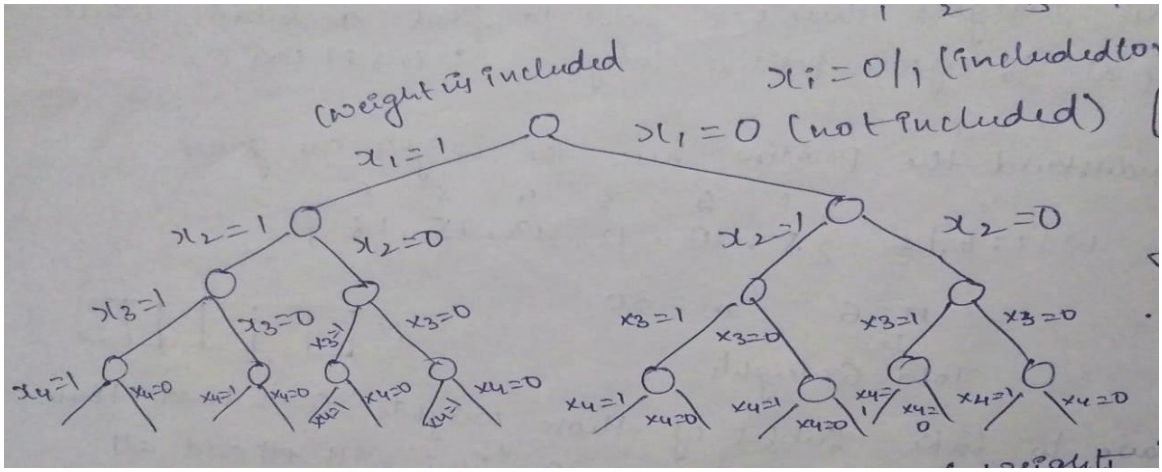
Sum of Subsets:

Let S = { S1, S2, S3,….., Sn } be a set of 'n' positive integers then we have to find a subset where sum is equals to given positive integers D or m or C.

Let us understand the problem here the weights are given

W[1:6] = { 5, 10, 12, 13, 15, 18 } n = 6(total 6 weights) m= 30

- We have to take subset of those weights such that there sum(total) is exactly m=30. If I will add all of the weights it doesn't be 30 more than 30.
- Whichever the weights that I am include I write them as solution and this solution will be either in 0/1 form. (Whether it includes or not).
- Now, how to solve the problem. So, the solution of those weights can be done in various ways.
- So, let us try all those possible ways of their selection and find out which of those selection is giving us this exact weight **M = 30.**



Like there are 6 weights I will be getting 7 levels including the root and the height will be '6'.

So total how many paths are making $2^6 = 2^n$ it's exponential and time consuming. So, the amount of time taken for solving this one is 2^n because that many paths are generating.

When we apply Backtracking we try to kill the nodes if they are not satisfying the Bounding Function.

Conditions: i) $\sum_{i=1}^{k} \text{wixi} + \text{wk} + 1 \leq \text{m}$

So, sum of the weights included so far till k^{th} object + weight of the next object should be less than or equal to m. If it is greater no use to add so, kill the node.

ii) $\sum_{i=1}^{k} \text{wixi} + \sum_{i=k+1}^{n} \text{wi} > \text{m}$

So far what all you have included and remaining weights i takes values from k+1 to n should be greater than m.