Name of Faculty: Alka Gulati

Designation: Associate Professor

Department: MCA

Subject: Design and Analysis of Algorithm

Unit: IV

Topic: Backtracking

Backtracking is a method for searching a set of solutions or find an optimal solution which satisfies some given constraint to a problem.

In backtracking method the solution set can be represented by an n tuple $(x_1,x_2,\ldots\ldots,x_n)$, where x_i are chosen from some finite set S_i.

This method can be used for optimization problem to find one or more solution vector that maximize or minimize or satisfy a given criterion function $C(x_1, x_2,\ldots,x_n)$.

Back tracking method at each step forms a partial solution set $(x_1, x_2,\ldots\ldots,x_i)$ and check it if this has any chances to find a solution

depending upon the criterion function C.

Solution Space in Backtracking

Backtracking method determine solution of a problem by searching for the solution set in the solution space. This searching can be organized in a tree called state space tree. **Backtracking method do depth first search of a state space tree.**

Terminologies

1) Problem state- Each node in the tree can be defined as problem state.
2) State space -A path from root to any other node defines a partial solution vector, can be called as state space.
3) Solution state- A solution state is a node s for which each node from root node to node s together can represent a tuple in solution set.
4) Answer states- Answer states are solution state which satisfies an implicit constraint.
5) State space tree- In state space tree problem state are generated from root node and then generated other nodes.
6) Live node - A live node is a generated node, for which all of its children node have not yet generated.
7) E-node - A E-node (Expanded node) is a live node, whose children are currently being generated.
8) Dead node - A dead node is that , which is not expanded further and all of its children is generated.

9) Bounding functions - Bounding functions are used to bound the searching in the tree.

Depth first node generated with bounding function is called backtracking.

Applications of Backtracking:-

i. 4 Queen's Problem
ii. 8 Queen's Problem
iii. Knapsack problem
iv. Graph Coloring Problem

In our syllabus ii and iii are there.

But before explaining 8 Queen's problem I will discuss 4 Queen's Problem .

4 Queen's Problem

Given chess board of field 4 x 4. The 4-Queen problem is to place 4-Queen on the chess board, so that no two Queen can "attack" each other. A Queen can attack vertically, horizontally and diagonally. In chess, queens can move all the way horizontally, vertically or diagonally (if there is no other queen in the way).But, no two Queen can attack each other. So, due to this restriction, each queen must be on a different row and column.

Backtracking strategy for 4-Queen problem is as follows

1. Let us, in the chess board rows and columns are numbered from 1 to 4 and also queens are numbered from 1 to 4.
2. Without loss of generality, assume that ith queen can be placed in ith row, because no two queen can place in the same row.

3. All solution can represented as 4-tuple (x_1, x_2, x_3, x_4), where x_i is the column number of the i th row of ith queen placed.
4. Here explicit constraints are $S_i = \{1, 2, 3, 4\}$, $1 \leq i \leq 4$ and the solution space will consist of 4 -tuple.
5. According to the implicit constraints no two queen can on the same row.
6. So, all solution are permutation of 8-tuple(1, 2, 3, 4)

Let two queen are placed in position (m, n) and (x, y)

Then two queens can placed diagonally if $m - n = x - y$ (1)

$m + n = x + y$ (2)

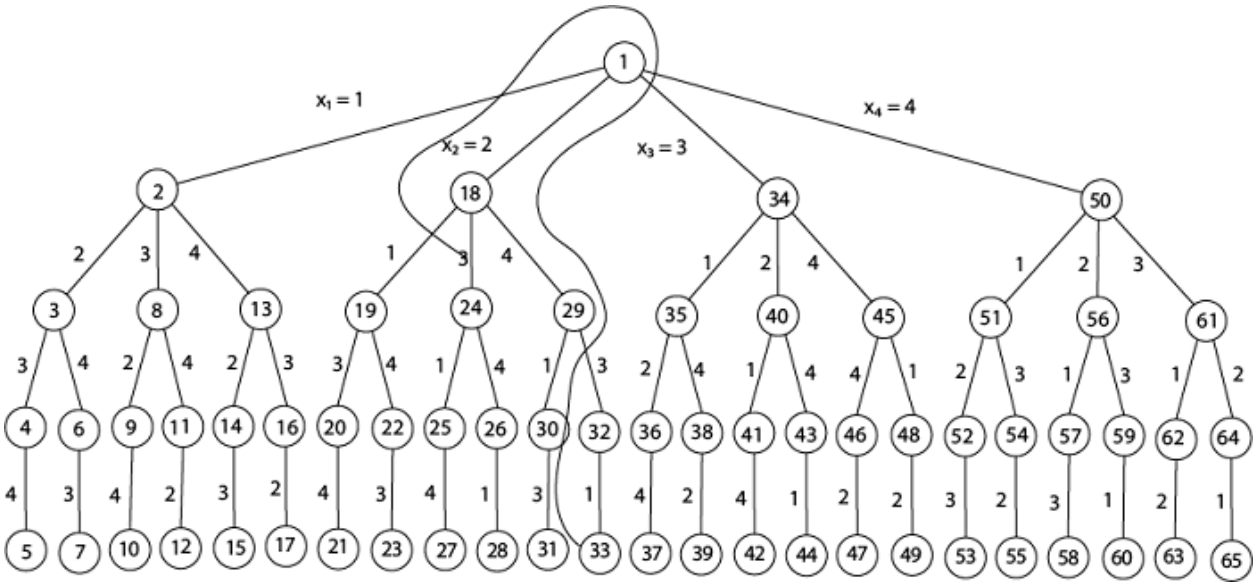
Therefore two queens can be on the same diagonal if and only if

$| n - y | = | m - x |$ This is an another implicit constraint .

The state space tree generated by 4-queen problem is as follows.

Here node at level i represent ith queen placed at ith row. i.e at level 1 it represents as 1st queen places in 1st row. Xi in ith level represents column number of ith queen placed in ith row. i.e x2 = 3 at level 2 means 2nd queen is in 3rd column of the 2nd row. Nodes are generated in depth first search manner. The path from root to leaf will represent a tuple in solution space.

All tuples are distinct and some tuples may not lead to a feasible solution.



State space tree

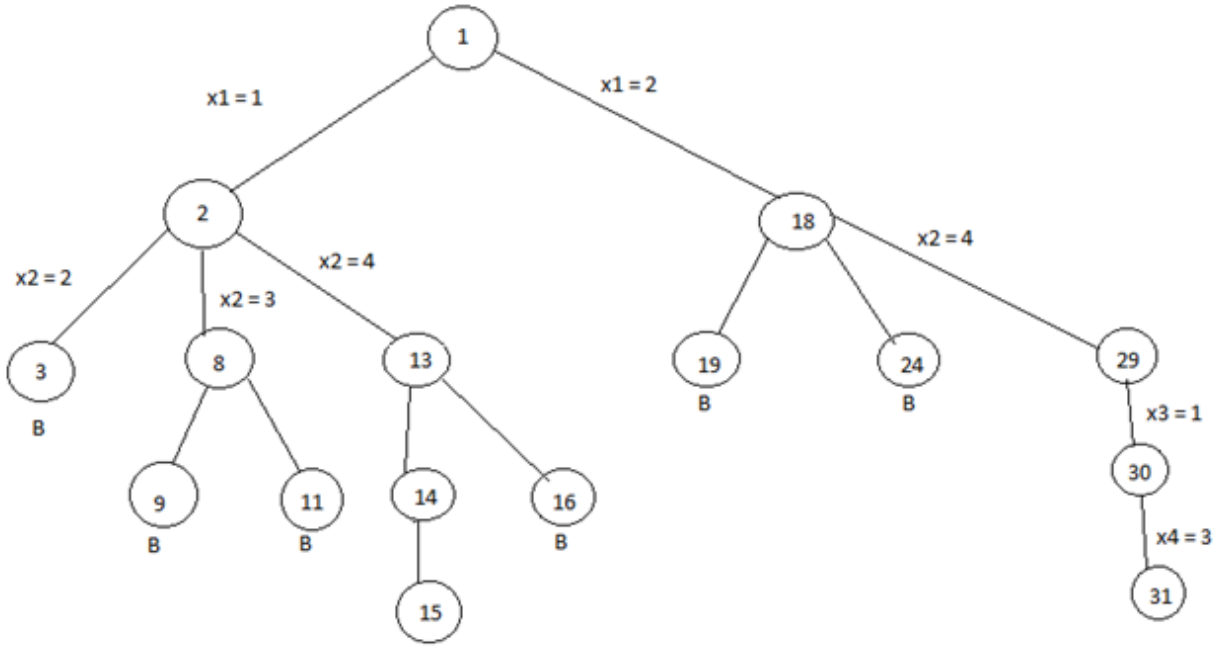
Now by using backtracking method we can bound the search of the state space tree using some constraint so that searching require less time. For this problems to bound a node n constraints or bounding conditions B(n) are:-

1. No two queens can place in same row i.e x_i always represents ith queen is in ith row.

2. No two queen in same column i.e values of x_i's are always distinct.

3. For two queen placed in (m, n) and (x, y) position on the board, value of $| n - y |$ cannot same as $| m - x |$.

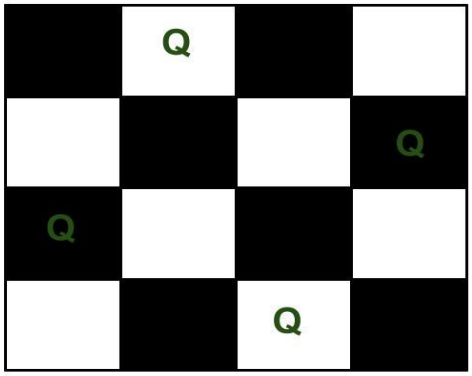
When a node is bounded using bounding condition it will not generate any nodes in its sub-tree because nodes in its sub-tree will not give a feasible solution any more.



Here node 3 is bounded because at level 1, x_1=1 means first time 1st queen is placed in 1st row, 1st column i.e position is (1,1) At level 2 , x2=2 means second time 2nd queen is placed in 2nd row, 2nd column i.e position(2,2).

Thus they will place in diagonally. It will violet the implicit constraint or bounding condition. So this combination cannot give a feasible solution any more. So, the children of node 3 will not generated further. Hence node 3 will bound. Here is a path from root 1 to leaf 3 and this will generate one feasible solution set (2, 4, 1, 3) where x_1=2, x_2=4, x_3=1, x_4=3.

Position of the 4 queens are (1,2), (2,4), (3,1) and(4,3) respectively.



Other solution is : - {(1,3) , (2,1) (3,4) ,(4,2)}