SNS COLLEGE OF TECHNOLOGY, COIMBATORE –35

Coping with the Limitations Backtracking: n Queens problem

Coping with the Limitations

- some problems that are difficult to solve algorithmically
- At the same time, few of them are so important, we must solve by some other technique
- Two algorithm design techniques
  - Backtracking
  - branch-and-bound

Coping with the Limitations

- backtracking and branch-and-bound are based on the construction of a state-space tree
- Both techniques terminate a node as soon as it can be guaranteed that no solution to the problem
- few approximation algorithms for solving the
  - Assignment Problem
  - traveling salesman
  - knapsack problems
- Algorithms for Solving Nonlinear Equations
  - Bisection Method
  - False Position Method
  - Newton's Method

Exact Solution Strategies

- Exhaustive search (brute force)
  - useful only for small instances
- Dynamic programming
  - applicable to some problems (e.g., the knapsack problem)
- Backtracking
  - eliminates some unnecessary cases from consideration
  - yields solutions in reasonable time for many instances but worst case is still exponential
- Branch-and-bound
  - further refines the backtracking idea for optimization problems

Coping with the Limitations of Algorithm Power

- Backtracking
  - n-Queens Problem
  - Hamiltonian Circuit Problem
  - Subset-Sum Problem
- Branch-and-Bound
  - Assignment Problem
  - Knapsack Problem
  - Traveling Salesman Problem
- Approximation Algorithms for NP-Hard Problems
  - Approximation Algorithms for the Traveling Salesman Problem
  - Approximation Algorithms for the Knapsack Problem

Solving Difficult Combinatorial Problems

- Two principal approaches to dealing with difficult combinatorial problems(i.e.,NP-hard problems)
- Use a strategy that guarantees solving the problem exactly but doesn’t guarantee to find a solution in polynomial time
- Use an approximation algorithm that can find an approximate(sub- optimal) solution in polynomial time

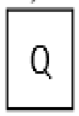
Backtracking

- Construct the *state-space tree*
  - nodes: partial solutions
  - edges: choices in extending partial solutions
- Explore the state space tree using depth-first search (dfs)
- "Prune" *nonpromising nodes*
  - dfs stops exploring subtrees rooted at nodes that cannot lead to a solution and backtracks to such a node's parent to continue the search

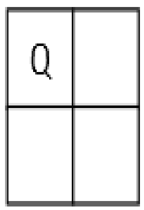
N-Queen Problem

The problem is to place n queens on an $n \times n$ chessboard so that no two queens attack each other by being in the same row or in the same column or on the same diagonal.

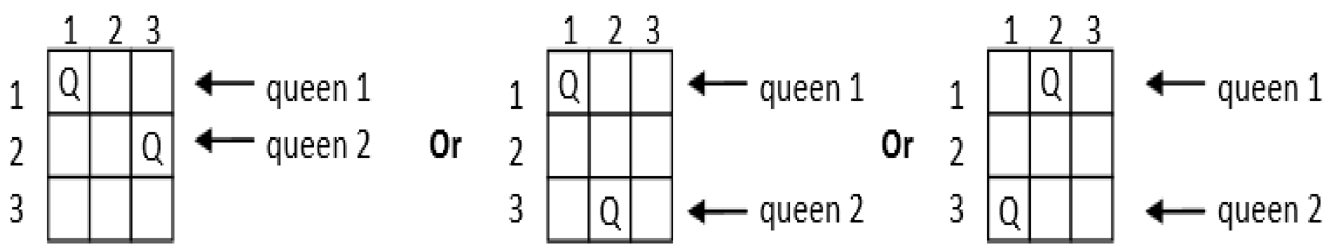
For n = 1, the problem has **a trivial solution.**



For n = 2, it is easy to see that there is **no solution** to place 2 queens in 2 × 2 chessboard.

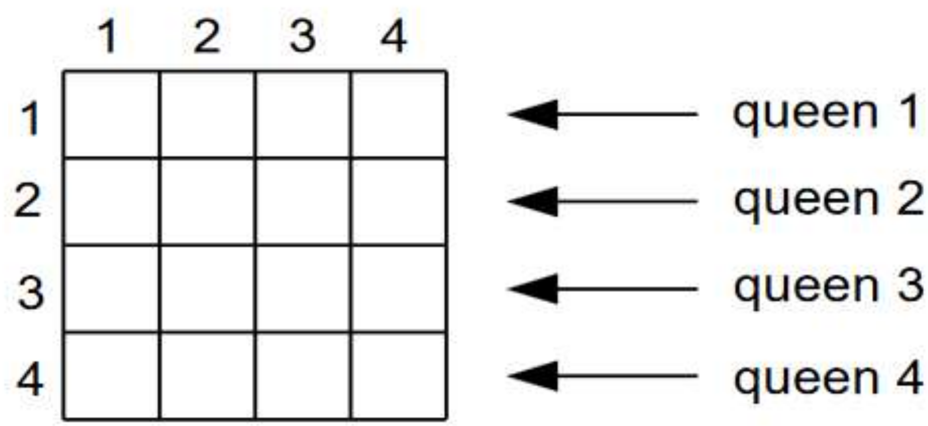


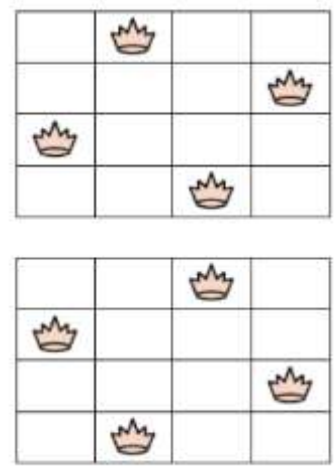
For n = 3, it is easy to see that there is **no solution** to place 3 queens in 3 × 3 chessboard.



N-Queen Problem

- **The *n*-queens problem**: is to place *n* queens on an *n*-by-*n* chess board so that no two queens attack each other by being in the same row, or in the same column, or on the same diagonal.
- Solution $x = (x_1, x_2, x_3, x_4) = (2, 4, 1, 3)$

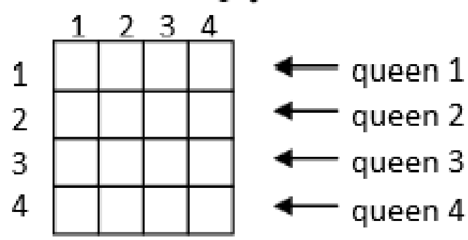




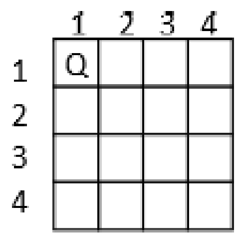
4-Queen Problem

For *n* = 4, There is **solution** to place 4 queens in 4 × 4 chessboard. the four-queens problem solved by the backtracking technique.

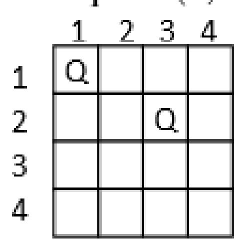
Step 1: Start with the empty board



Step 2: Place queen 1 in the first possible position of its row, which is in column 1 of row 1.



Step 3: place queen 2, after trying unsuccessfully columns 1 and 2, in the first acceptable position for it, which is square (2, 3), the square in row 2 and column 3.



4-Queen Problem

Step 4: This proves to be a dead end because there is no acceptable position for queen 3. So, the algorithm backtracks and puts queen 2 in the next possible position at (2, 4).

	1	2	3	4
1	Q			
2				Q
3				
4				

Step 5: Then queen 3 is placed at (3, 2), which proves to be another dead end.

	1	2	3	4
1	Q			
2				Q
3		Q		
4				

Step 6: The algorithm then backtracks all the way to queen 1 and moves it to (1, 2).

	1	2	3	4
1		Q		
2				
3				
4				

4-Queen Problem

Step 7: The queen 2 goes to (2, 4).

	1	2	3	4
1		Q		
2				Q
3				
4				

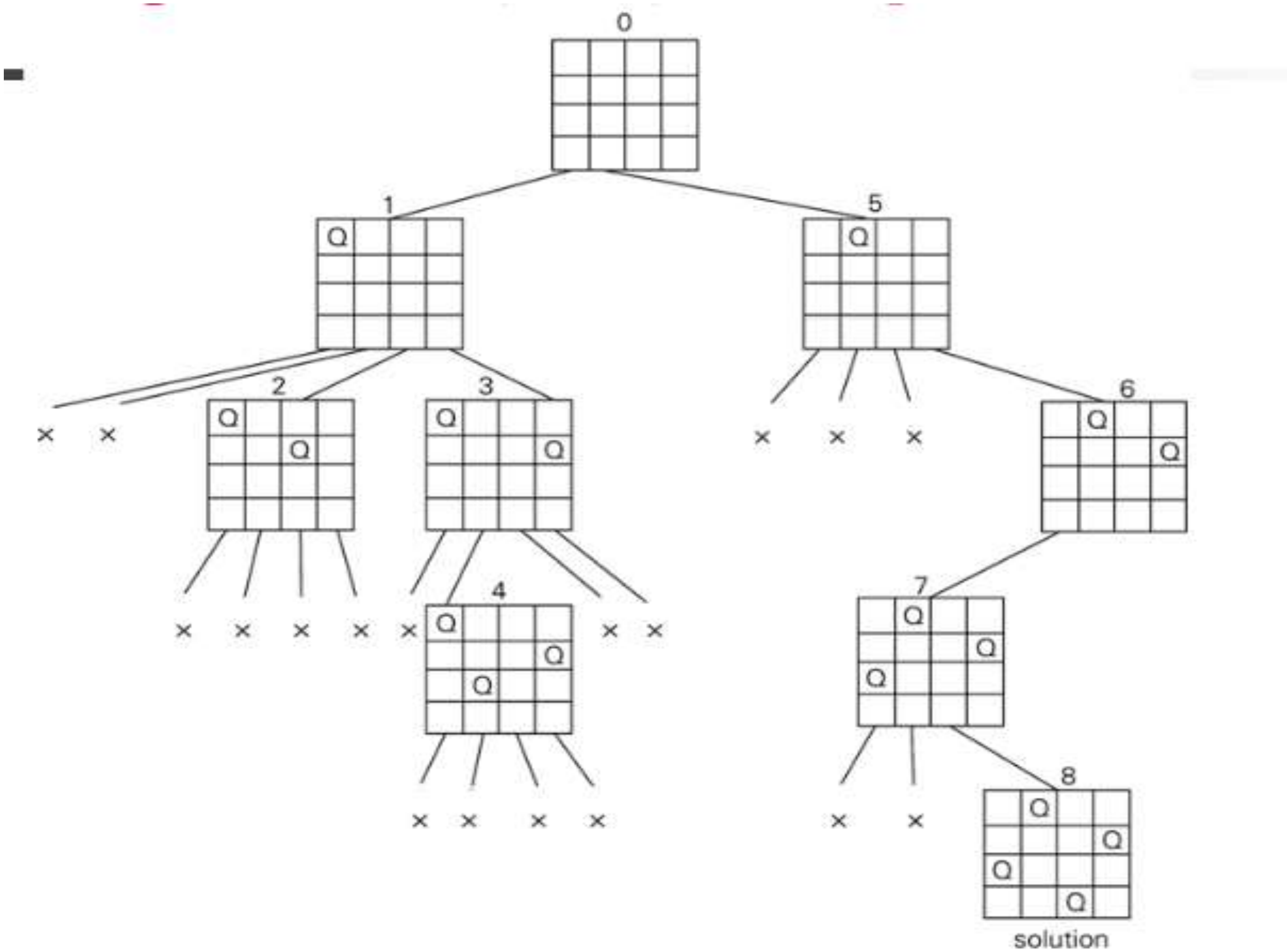
Step 8: The queen 3 goes to (3, 1).

	1	2	3	4
1		Q		
2				Q
3	Q			
4				

Step 9: The queen 3 goes to (4, 3). This is a solution to the problem.

	1	2	3	4
1		Q		
2				Q
3	Q			
4			Q	

Solution for 4 Queens Problem



Solutions: (2, 4, 1, 3) and (3, 1, 4, 2) (reflection)

State Space Tree of 8 Queens Problem

For n **= 8**, There is **solution** to place 8 queens in 8 × 8 chessboard.

	1	2	3	4	5	6	7	8
1				Q				
2						Q		
3								Q
4			Q					
5	Q							
6							Q	
7					Q			
8		Q						