UNIT V

BACKTRACKING AND BRANCH-AND-BOUND

Backtracking – N-Queens Problem – Hamiltonian Circuit Problem – Subset Sum Problem – Branch-and- Bound – Travelling Salesman Problem

Tackling Difficult Combinatorial Problems

There are two principal approaches to tackling difficult combinatorial problems (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

Exact Solution Strategies

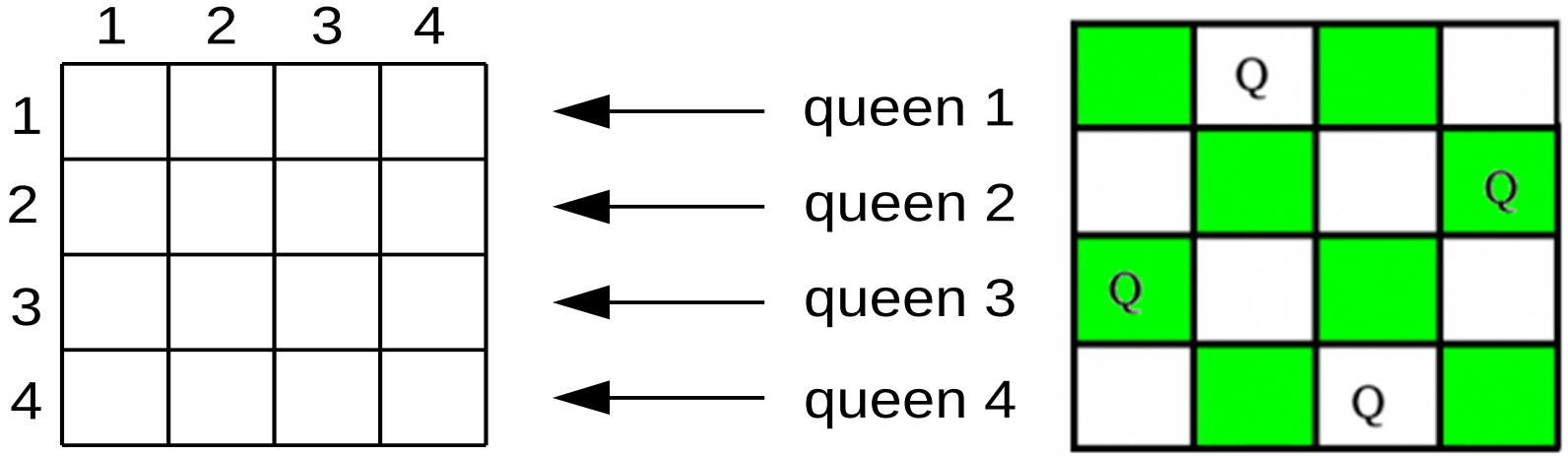
- *exhaustive search* (brute force)
  - useful only for small instances
- *dynamic programming*
  - applicable to some problems (e.g., the knapsack problem)
- *backtracking*
  - eliminates some unnecessary cases from consideration
  - yields solutions in reasonable time for many instances but worst case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

Backtracking

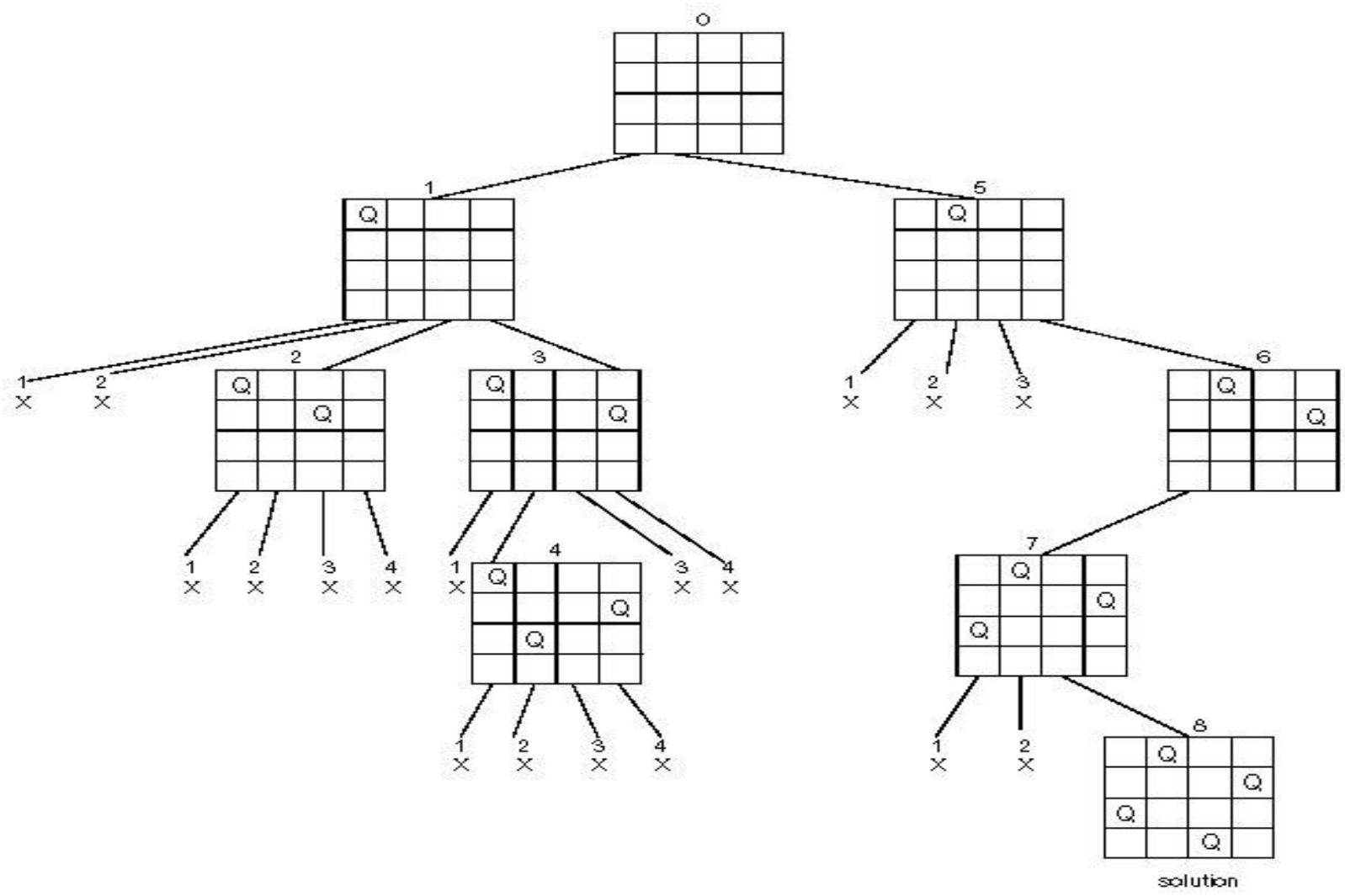
- Construct the *state-space tree*
  - nodes: partial solutions
  - edges: choices in extending partial solutions
- Explore the state space tree using depth-first search
- "Prune" *non-promising nodes*
  - dfs stops exploring subtrees rooted at nodes that cannot lead to a solution and backtracks to such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

Example: *n*-Queens Problem

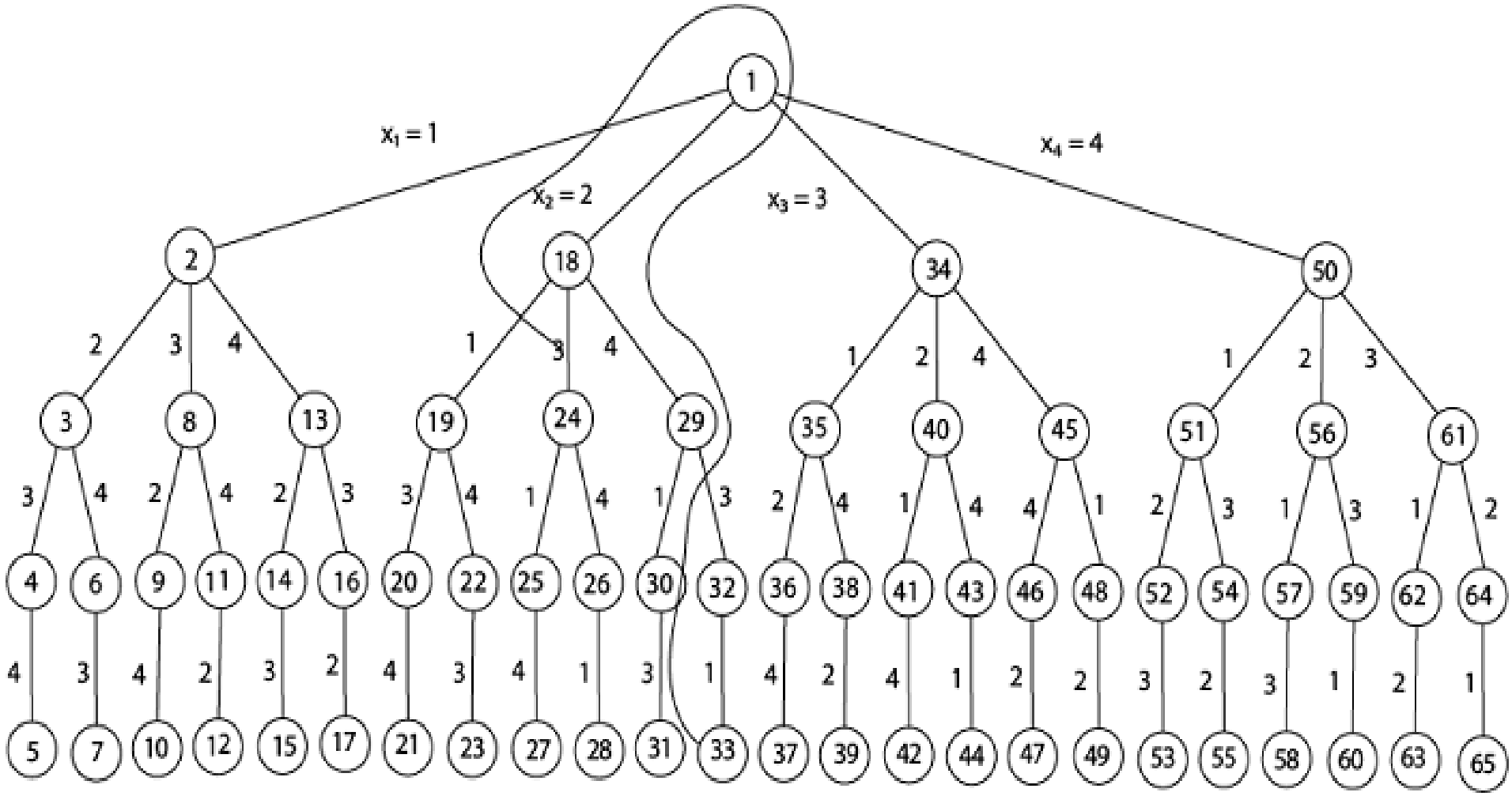
Place *n* queens on an *n*-by-*n* chess board so that no two of them are in the same row, column, or diagonal



State-Space Tree of the 4-Queens Problem



State-Space Tree of the 4-Queens Problem



8-queen Problem

Number of queens N =8 and Queens: Q1,Q2….Q8

			Q1				
					Q2		
							Q3
	Q4						
						Q5	
Q6							
		Q7					
				Q8			

Fig.: Solution space table for 8-queens

Hence solution vector for 8 queens is (4,6,8,2,7,1,3,5).


Thank You