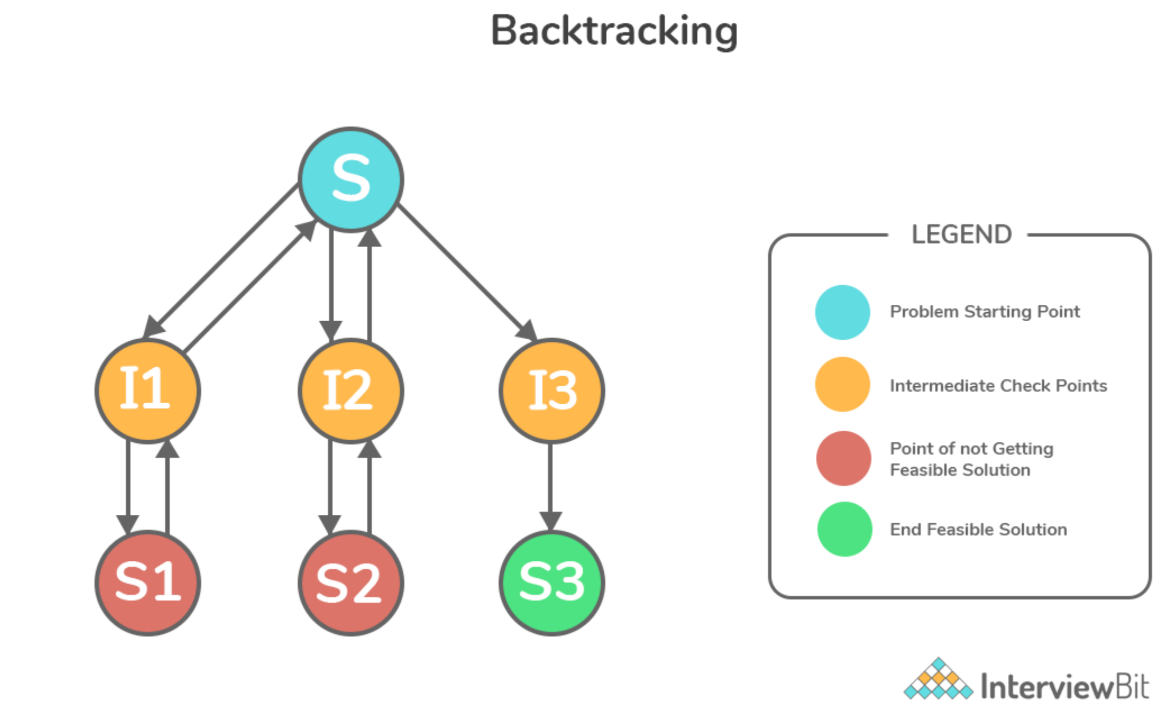
Lecture 22: Backtracking Algorithms

Date: 11/02/2023



reference: https://crackfaang.medium.com/backtracking-with-c-91e3bfc56a21

Find All Permutations

Given an array of numbers, find all possible ways to represent the same array. In other words, find all possible ways to rearrange the elements in the array.

Examples,

Input: [1,2,3]
Output: [1,2,3], [1,3,2], [2,1,3], [2,3,1], [3,1,2], [3,2,1]

Input: [5,2]
Output: [2,5], [5,2]

IN CLASS

Find All Subsets

Given an array of numbers, find all possible subsets of that array.
In other words, find all possible contiguous segments of the array.

Examples,
Input: [1,2,3]
Output: [1], [2], [3], [1,2], [2,3], [1,2,3]

Input: [1,2]
Output: [1], [2], [1,2]

IN CLASS

N-Queen Problem

Given a number n, find all possible queen positions in n*n board such that, none of the queens can kill each other.

IN CLASS

Generic OOP Pseudocode

```
class Problem{
    bool is_goal(State state);
    vector<Move> get_possible_moves(State state);
}
```

```
void backtrack(state){
    if Problem.is_goal(state)
        // do something with goal
        return;

    all_possible_moves = Problem.get_possible_moves(state)

    for move in all_possible_moves:
        move.apply(state)
        backtrack(state)
        move.undo(state)
}
```