Backtracking

Dr. Anant Ram
Associate Professor
Dept. of CEA, GLA University, Mathura

Backtracking

- Is used to solve problems for which a sequence of objects is to be selected from a set such that the sequence satisfies some constraint.
- Traverses the state Tree using a depth-first search with pruning.
- Performs a depth-first traversal of a tree.
- Continues until it reaches a node that is non-viable or non-promising.
- Prunes the sub tree rooted at this node and continues the depth-first traversal of the tree.
- This gives a significant advantage over an exhaustive search of the tree for the average problem.
- Worst case: Algorithm tries every path, traversing the entire search space as in an exhaustive search.

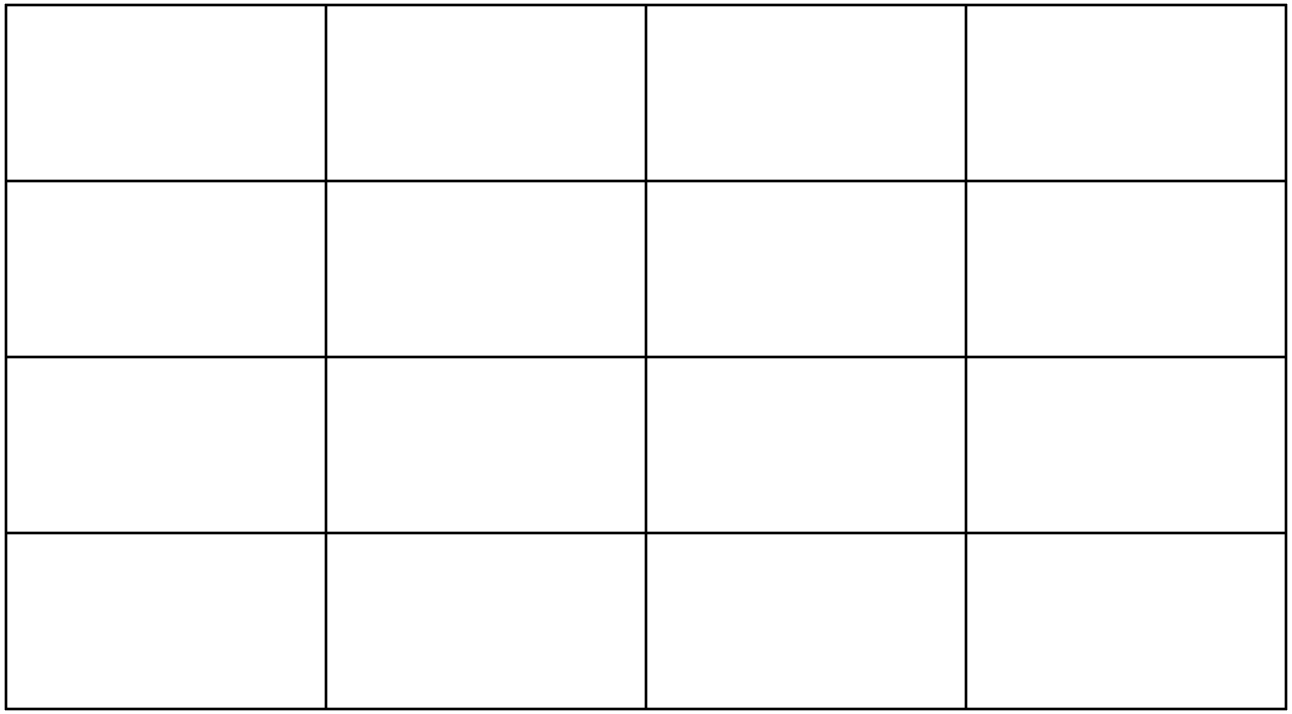
Sum of subsets

- Problem: Given n positive integers $w_1, \ldots w_n$ and a positive integer S. Find all subsets of $w_1, \ldots w_n$ that sum to S.

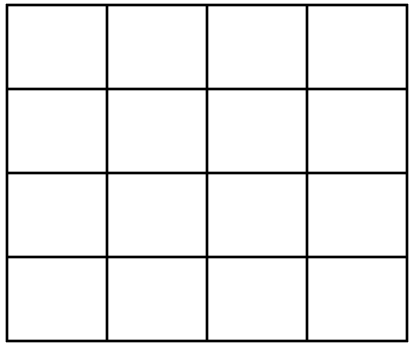
- $w_1 = 3, w_2 = 4, w_3 = 5, w_4 = 6, w_5 = 7;\ S = 13$

$$w_1 = 3,\ w_2 = 4,\ w_3 = 5,\ w_4 = 6,\ w_5 = 7;\ \ S = 13$$

N-Queen Problem



N-Queen Problem



Graph Coloring Problem

Let G be undirected graph and let c be an integer.

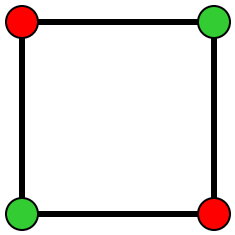
Assignment of colors to the vertices or edges such that no two adjacent vertices are to be similarly colored.

We want to minimize the number of colors used.

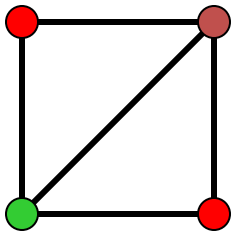
The smallest c such that a c-coloring exists is called the graph's chromatic number and any such c-coloring is an optimal coloring.

Coloring of Graph

1. The graph coloring optimization problem: find the minimum number of colors needed to color a graph.
2. The graph coloring decision problem: determine if there exists a coloring for a given graph which uses at most m colors.



Two colors



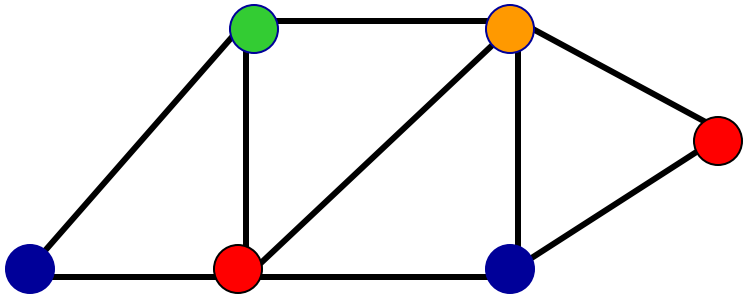
No solution with two colors

Coloring of Graphs

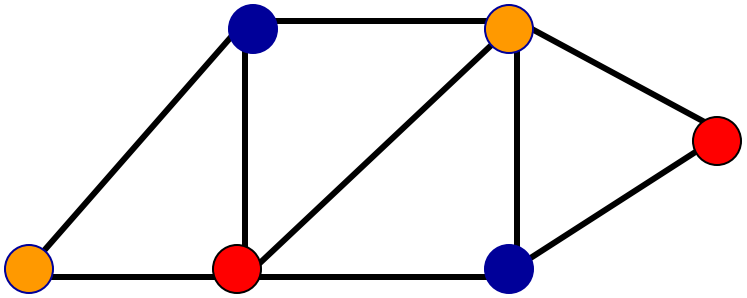
Practical applications: scheduling, time-tabling, register allocation for compilers, coloring of maps.

A simple graph coloring algorithm - choose a color and an arbitrary starting vertex and color all the vertices that can be colored with that color.

Choose next starting vertex and next color and repeat the coloring until all the vertices are colored.



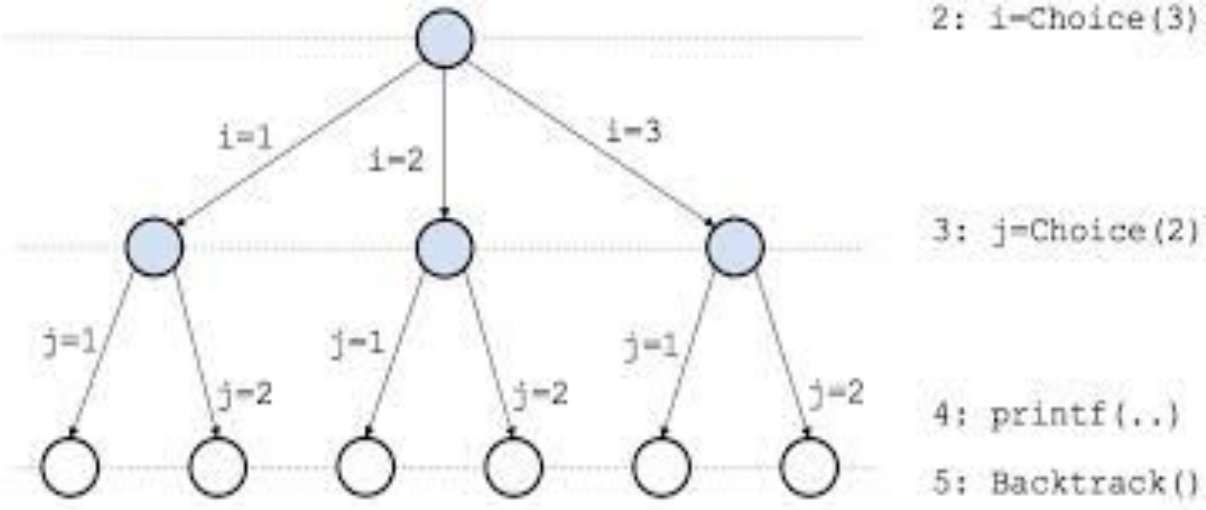
Four colors



Three colors are enough

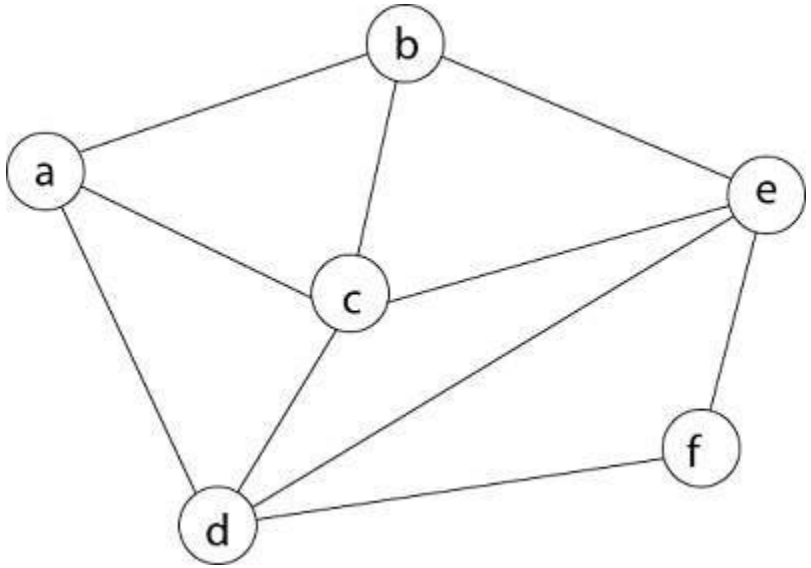
Backtracking

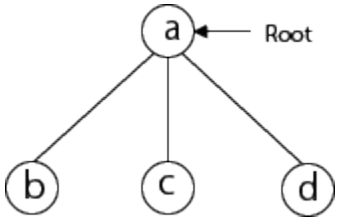
- Partial 3-coloring (3 colors) is solved by the following method:
- Color first vertex with 1st color, color next vertex with the next color, check if those two vertices are adjacent, if not - coloring is legal, proceed to next vertex, if yes and color is the same – coloring is illegal, try next color for second vertex. If all colors tried and all colorings are illegal, backtrack, try next color for previous vertex etc.
- Note: sometimes solution is impossible.
- Exponential O(3^n) complexity is reduced to O(n) on average.

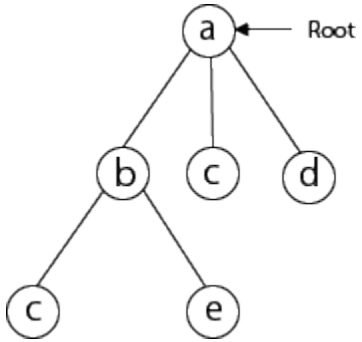


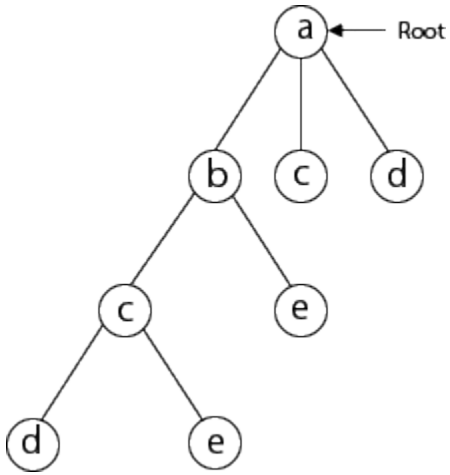
Hamiltonian Circuit Problems

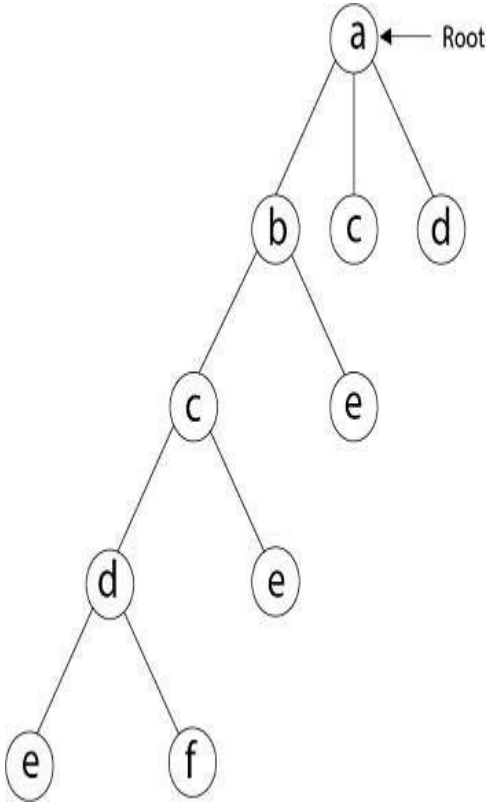
- Given a graph G = (V, E) , A **Hamiltonian cycle** is a closed loop on a graph where every node (vertex) is visited exactly once.

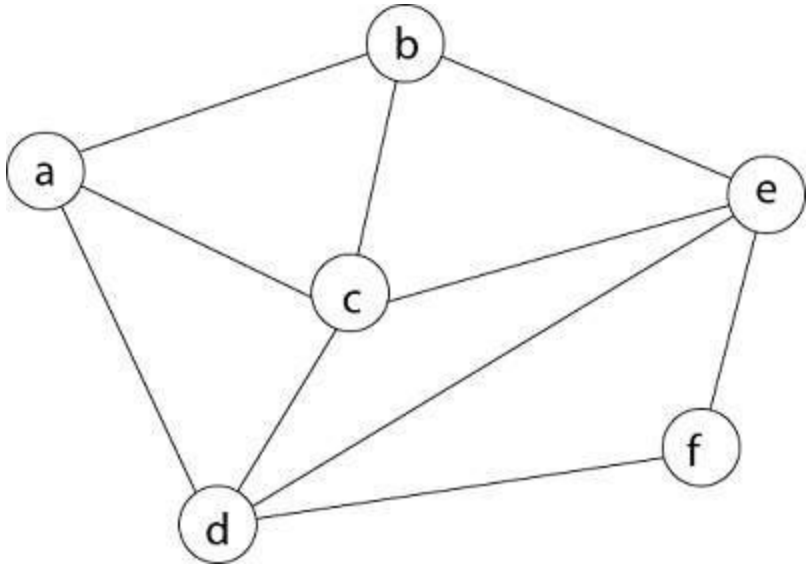


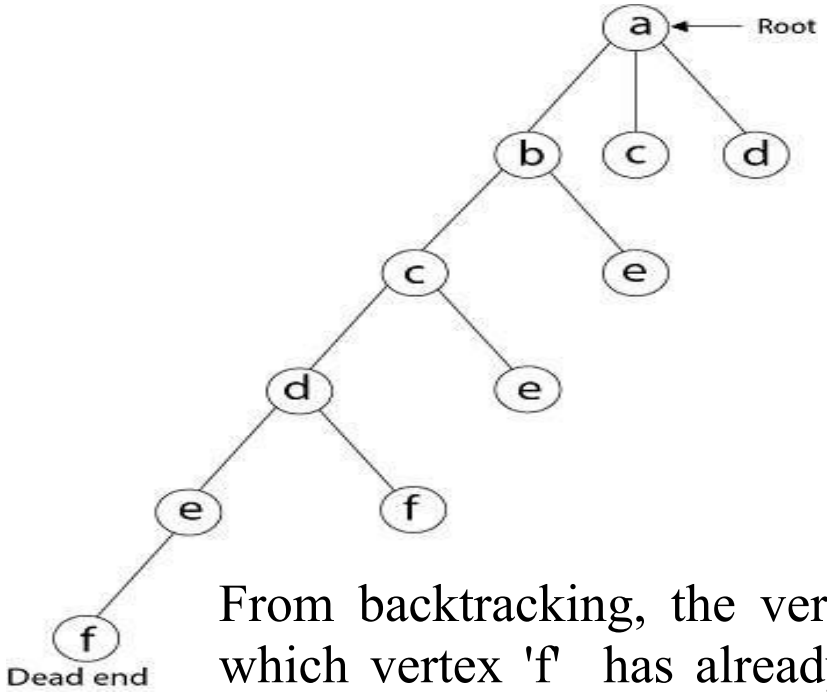


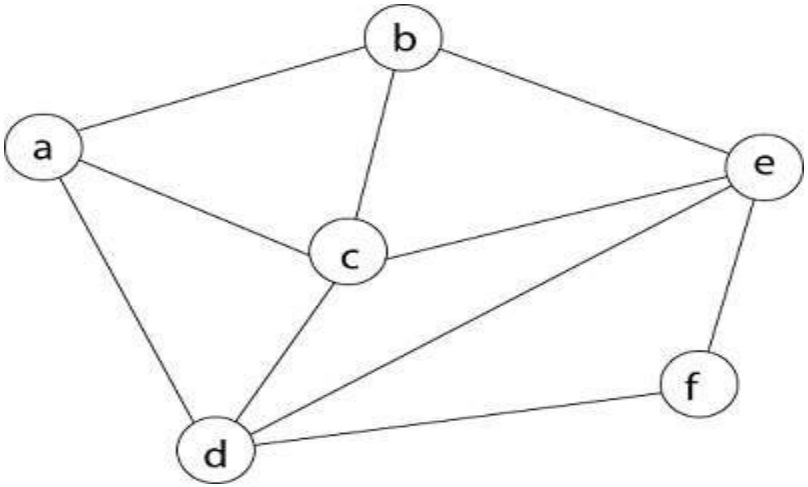












From backtracking, the vertex adjacent to 'e' is b, c, d, and f from which vertex 'f' has already been checked, and b, c, d have already visited. So, again we backtrack one step. Now, the vertex adjacent to d are e, f from which e has already been checked, and adjacent of 'f' are d and e. If 'e' vertex, revisited them we get a dead state. So again we backtrack one step.

Now, adjacent to c is 'e' and adjacent to 'e' is 'f' and adjacent to 'f' is 'd' and adjacent to 'd' is 'a.' Here, we get the Hamiltonian Cycle as all the vertex other than the start vertex 'a' is visited only once. (a - b - c - e - f -d - a).