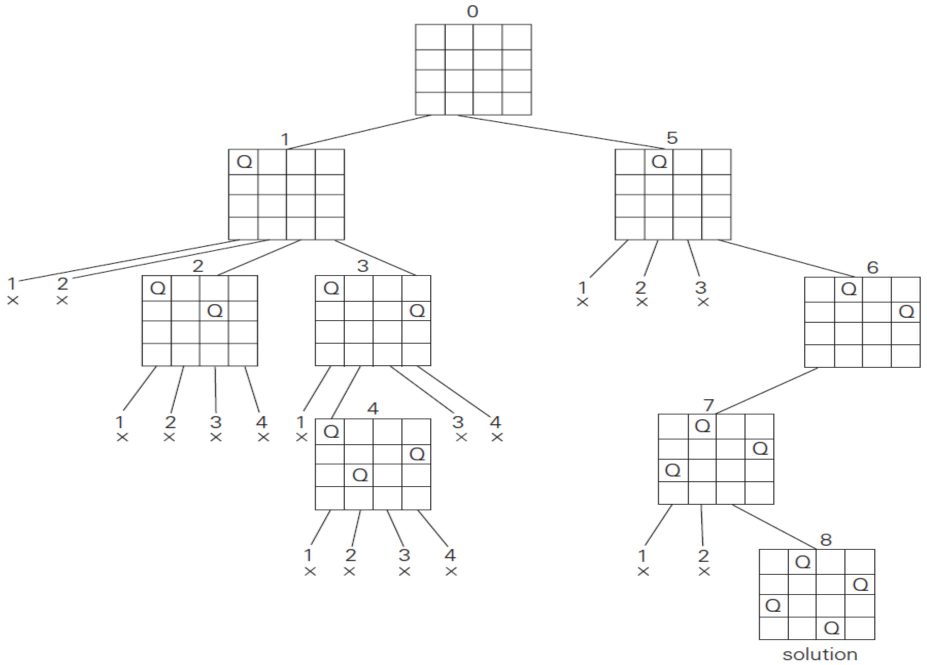
BACKTRACKING

- Backtracking is an algorithmic-technique for solving problems recursively by trying to build a solution incrementally, one piece at a time, removing those solutions that fail to satisfy the constraints of the problem at any point of time. Eg: SudoKo solving Problem
- This kind of processing can very well be implemented using a ***state-space tree***.
- Its root represents an initial state before the search for a solution begins.
- The nodes of the first level in the tree represent the choices made for the first component of a solution; the nodes of the second level represent the choices for the second component, and so on.
- A node in a state-space tree is said to be ***promising*** if it corresponds to a partially constructed solution that may still lead to a complete solution; otherwise it is called ***nonpromising.***
- Leaves represent either nonpromising dead ends or complete solutions found by the algorithm.
- A state-space tree for a backtracking algorithm is constructed in the manner of depth first search.
- If the current node is promising, its child is generated by adding the first remaining legitimate option for the next component of a solution, and the processing moves to this child.
- If the current node turns out to be nonpromising, the algorithm backtracks to the node’s parent to consider the next possible option for its last component.

N-Queens Problem

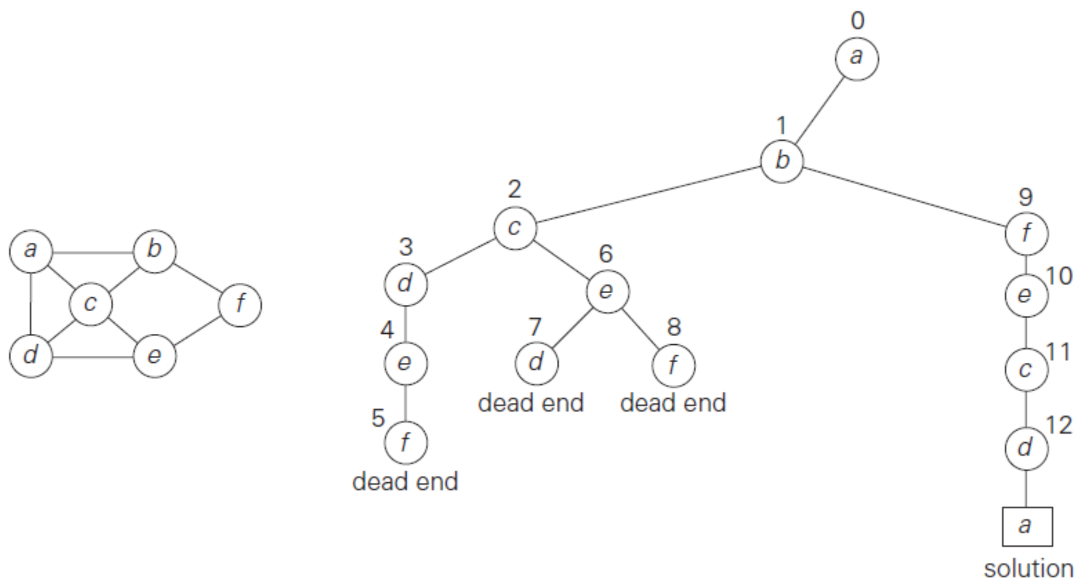
- The problem is to place *n* queens on an $n \times n$ chessboard so that no two queens attack each other by being in the same row or in the same column or on the same diagonal.
- Let us consider the four-queens problem and solve it by the backtracking technique:
  - ✓ Start with the empty board and then place queen 1 in the first possible position of its row, which is in column 1 of row 1.
  - ✓ Then we 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.
  - ✓ 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).

- ✓ Then queen 3 is placed at (3, 2), which proves to be another dead end.
- ✓ The algorithm then backtracks all the way to queen 1 and moves it to (1, 2).
- ✓ Queen 2 then goes to (2, 4), queen 3 to (3, 1), and queen 4 to (4, 3), which is a solution to the problem.
- ✓ The state-space tree of this problem is shown in below figure:



Hamiltonian Circuit Problem

- Hamiltonian path in an undirected graph is a path that visits each vertex exactly once.
- Hamiltonian circuit problem determines whether a given graph contains Hamiltonian cycle or not. If it contains, then it should print the path.
- An example problem is explained below.

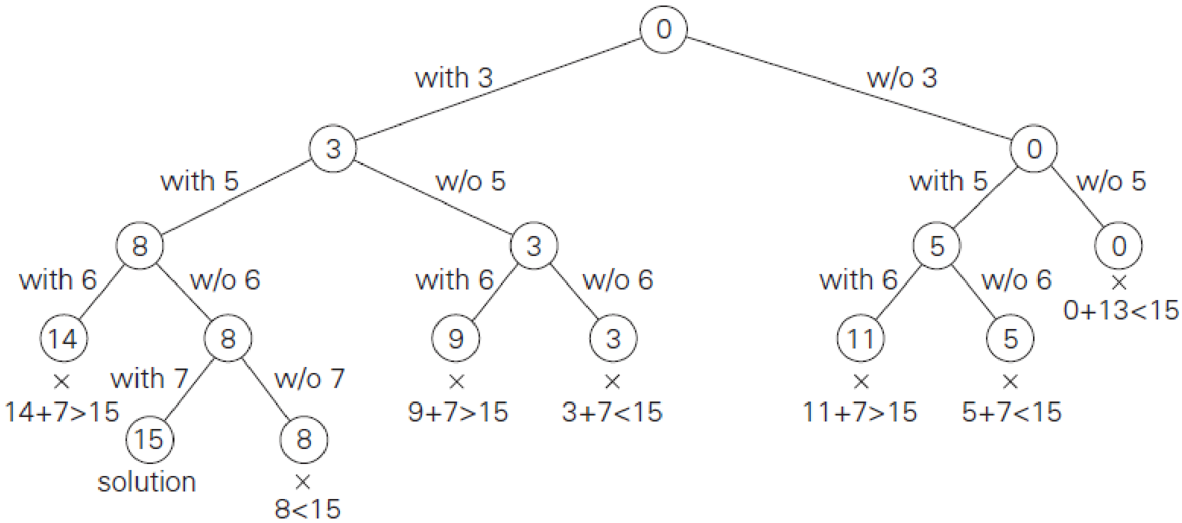


- ✓ Let ‘a’ be the first node in the path.
- ✓ Vertex ‘*b*’ can be selected next.
- ✓ From ‘*b*’, the algorithm proceeds to ‘*c*’, then to ‘*d*’, then to ‘*e*’, and finally to ‘*f*’, which proves to be a dead end.
- ✓ So the algorithm backtracks from ‘*f*’ to *e*, then to *d*, and then to *c*, which provides the first alternative for the algorithm to pursue.
- ✓ Going from *c* to *e* eventually proves useless, and the algorithm has to backtrack from *e* to *c* and then to *b*.
- ✓ From there, it goes to the vertices *f* , *e*, *c*, and *d*, from which it can legitimately return to *a*, yielding the Hamiltonian circuit *a*, *b*, *f* , *e*, *c*, *d*, *a*.

- If we want to find another solution, the process could be backtracked from the leaf of the solution found.

Subset-Sum Problem

- The problem is finding a subset of a given set $A = \{a_1, . . . , a_n\}$ of n positive integers whose sum is equal to a given positive integer d.
- For example if A = {2, 3, 4, 5, 12}, sum = 9; the solutions are {{4,5}, {2,3,4}}.
- It is convenient to sort the set’s elements in increasing order.
- The state-space tree can be constructed as a binary tree as shown in below figure for the instance $A = \{3, 5, 6, 7\}$ and $d = 15$.



- ✓ The root of the tree represents the starting point, with no decisions about the given elements made as yet.
- ✓ Its left and right children represent, respectively, inclusion and exclusion of a_1 in a set being sought.

- ✓ Similarly, going to the left from a node of the first level corresponds to inclusion of a_2 while going to the right corresponds to its exclusion, and so on.
- ✓ Thus, a path from the root to a node on the *i*th level of the tree indicates which of the first *i* numbers have been included in the subsets represented by that node.
- ✓ The value of *s*, the sum of these numbers, is recorded in the node.
- ✓ If *s* is equal to *d*, the solution is reached.
- ✓ The problem can be stopped there or, if all the solutions need to be found, continue by backtracking to the node's parent.
- ✓ If *s* is not equal to *d*, we can terminate the node as nonpromising.