

# Branch and Bound Algorithms

The design technique known as branch and bound is very similar to backtracking in that it searches a tree model of the solution space and is applicable to a wide variety of discrete combinatorial problems. Branch-and-Bound algorithms can be applied only to optimization problems.

Each node in the combinatorial tree generated in the last Unit defines a problem state. All paths from the root to other nodes define the state space of the problem.

**Solution states** are those problem states 's' for which the path from the root to 's' defines a tuple in the solution space. The leaf nodes in the combinatorial tree are the solution states.

**Answer states** are those solution states 's' for which the path from the root to 's' defines a tuple that is a member of the set of solutions (i.e., it satisfies the implicit constraints) of the problem.

The tree organization of the solution space is referred to as the state space tree. A node which has been generated and all of whose children have not yet been generated is called a live node. The live node whose children are currently being generated is called the E-node (node being expanded).

A dead node is a generated node, which is not to be expanded further or all of whose children have been generated.

**Bounding functions** are used to kill live nodes without generating all their children.

Depth first node generation with bounding function is called backtracking. State generation methods in which the E-node remains the E-node until it is dead lead to branch-and-bound method. The term branch-and-bound refers to all state space search methods in which all children of the E-node are generated before any other live node can become the E-node. In branch-and-bound terminology breadth first search(BFS)- like state space search will be called FIFO (First In First Output) search as the list of live nodes is a first-in-first-out list(or queue).

A D-search(depth search) state space search will be called LIFO (Last In First Out) search, as the list of live nodes is a list-in-first-out list (or stack).

Bounding functions are used to help avoid the generation of sub trees that do not contain an answer node. The branch-and-bound algorithms search a tree model of the solution space to get the solution. However, this type of algorithms is oriented more toward optimization. An algorithm of this type specifies a real -valued cost function for each of the nodes that appear in the search tree.

Usually, the goal here is to find a configuration for which the cost function is minimized. The branch-and-bound algorithms are rarely simple.

## Difference between FIFO , LIFO and LC Branch and Bound


Branch & Bound discovers branches within the complete search space by using estimated bounds to limit the number of possible solutions. The different types (FIFO, LIFO, LC) define different 'strategies' to explore the search space and generate branches.

**FIFO** (first in, first out): always the oldest node in the queue is used to extend the branch. This leads to a **breadth-first** search, where all nodes at depth  $d$  are visited first, before any nodes at depth  $d+1$  are visited.

**LIFO** (last in, first out): always the youngest node in the queue is used to extend the branch. This leads to a **depth-first** search, where the branch is extended through every 1st child discovered at a certain depth, until a leaf node is reached.

**LC** (lowest cost): the branch is extended by the node which adds the lowest additional costs, according to a given cost function. The strategy of traversing the search space is therefore defined by the cost function.

## Popular Branch and Bound Programming Questions

- Job Assignment Problem
- Travelling salesman problem
- 0/1 Knapsack when item weights are not integers 

## Web links - to continue

- [Branch and Bound explanation](#)
- [Job Assignment Problem using Branch And Bound](#)
- [Branch And Bound—Why Does It Work?](#)