Iterative deepening depth-first search (IDDFS)

Iterative deepening depth-first search (IDDFS) is an algorithm that is an important part of an Uninformed search strategy just like BFS and DFS.

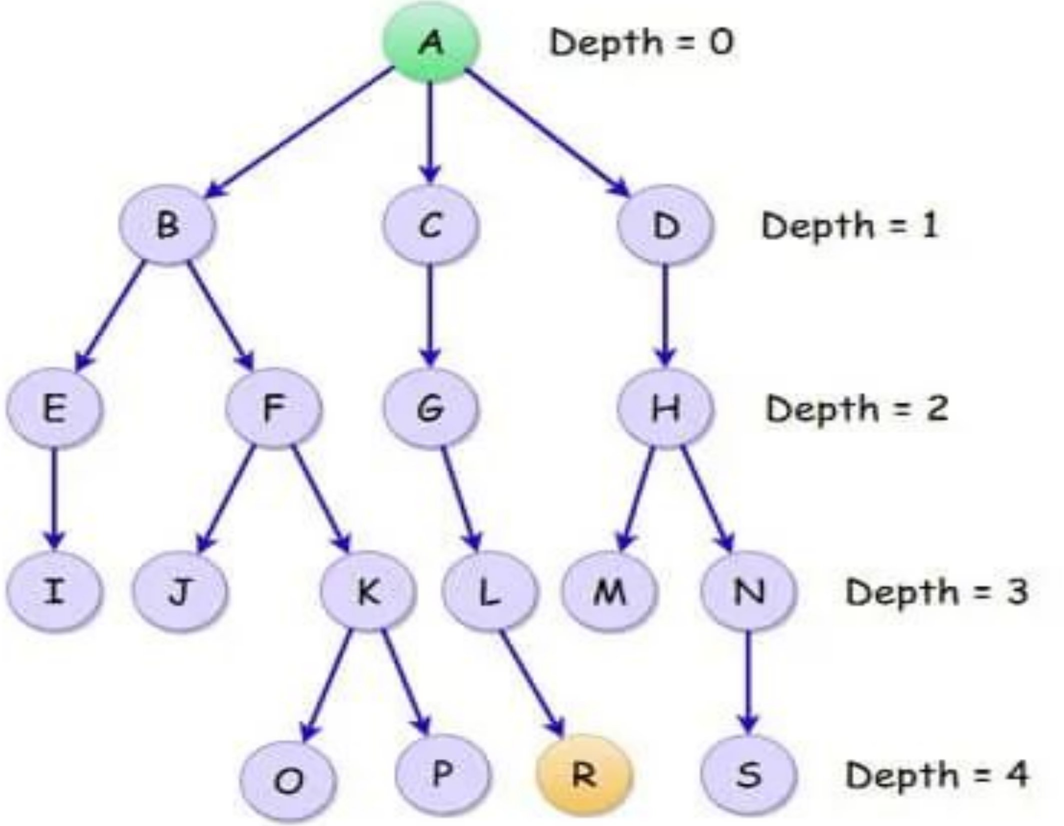
In IDDFS, We have found certain limitations in BFS and DFS so we have done hybridization of both the procedures for eliminating the demerits lying in them individually.

We do a limited depth-first search up to a fixed “limited depth”. Then we keep on incrementing the depth limit by iterating the procedure unless we have found the goal node or have traversed the whole tree whichever is earlier.

Algorithm

- Consider making a breadth-first search into an iterative deepening search.
- We can do this by having aside a DFS which will search up to a limit. It first does searching to a pre-defined limit depth to depth and then generates a route length1.
- This is done by creating routes of length 1 in the DFS way. Next, it makes way for routes of depth limit 2, 3 and onwards.
- It even can delete all the preceding calculation all-time at the beginning of the loop and iterate. Hence at some depth eventually the solution will be found if there is any in the tree because the enumeration takes place in order.

Example of Iterative Deepening Depth-First Search



Here in the given tree, the starting node is A and the depth initialized to 0.

The goal node is R where we have to find the depth and the path to reach it. The depth from the figure is 4.

In this example, we consider the tree as a finite tree, while we can consider the same procedure for the infinite tree as well.

We knew that in the algorithm of IDDFS we first do DFS till a specified depth and then increase the depth at each loop.

This special step forms the part of DLS or Depth Limited Search. Thus the following traversal shows the IDDFS search.

The tree can be visited as: A B E F C G D H

DEPTH = {0, 1, 2, 3, 4}

DEPTH LIMITS	IDDFS
0	A
1	A B C D
2	A B E F C G D H
3	A B E I F J K C G L D H M N
4	A B E I F J K O P C G L R D H M N S

Advantages

- IDDFS gives us the hope to find the solution if it exists in the tree.
- When the solutions are found at the lower depths say n, then the algorithm proves to be efficient and in time.
- The great advantage of IDDFS is found in-game tree searching where the IDDFS search operation tries to improve the depth definition, heuristics, and scores of searching nodes so as to enable efficiency in the search algorithm.
- Another major advantage of the IDDFS algorithm is its quick responsiveness. The early results indications are a plus point in this algorithm. This followed up with multiple refinements after the individual iteration is completed.

Disadvantages

- The time taken is exponential to reach the goal node.
- The main problem with IDDFS is the time and wasted calculations that take place at each depth.
- The situation is not as bad as we may think of especially when the branching factor is found to be high.
- The IDDFS might fail when the BFS fails. When we are to find multiple answers from the IDDFS, it gives back the success nodes and its path once even if it needs to be found again after multiple iterations. To stop the depth bound is not increased further.