☞ **Optimality**

The DLS is a non-optimal algorithm since the depth that is chosen can be greater than d ($l > d$). Thus DLS is not optimal if $l > d$.

☞ **Time complexity**

It is similar to DFS, i.e. $O(b^l)$, where 1 is the specified depth limit.

☞ **Space complexity**

It is similar to DFS, it is $O(b^l)$, where 1 is the specified depth limit.

☞ **Conclusion – DLS**

(i) DLS is not the case for uninformed search strategy.

(ii) DLS algorithm is used when we know the search domain, and there exists a prior knowledge of the problem and its domain.

(iii) There is little idea of the goal nodes depth.

(iv) The problem with depth-limited search is to set the value of *l* optimally, so as not to leave out any solution.

Also keep the time and space complexity to a minimum.

2.11 ITERATIVE DEEPENING SEARCH TECHNIQUE (IDS OR IDDFS)

UQ. Explain Iterative Deepening search algorithms based on performance measure with justification; complete, optimal, Time and Space complexity.

(Q. 4(a), Dec. 18, 10 Marks)

(1) Iterative deepening search or more specifically iterative deepening depth-first search (IDS or IDDFS) is a **state space/graph search strategy** in which a **depth-limited version** of depth-first search is run repeatedly with increasing depth limits until the goal is found.

(2) IDDFS is equivalent to breadth-first search, but uses much less memory; on each iteration, it visits the nodes in the search tree in the same order as depth-first search, but the cumulative order in which nodes are first visited is effectively breadth-first.

(3) DDFS combines depth-first search's space-efficiency and breadth-first search's completeness (When the branching factor is finite). It is optimal when the **path cost is a non-decreasing function of the depth of the node.**

(4) The **time complexity of IDDFS is O (b^d)** and its **space complexity is $O(b^d)$**, where **b** is the branching factor and **d** is the depth of the shallowest goal.

(5) Since iterative deepening, visits states multiple times, it may seem wasteful, but it turns out to be not costly, since in a tree most of the nodes are in the bottom level, so it does not matter much if the upper levels are visited multiple times.

2.11.1 IDDFS Algorithm

```
Function iterative-Deepening-search (problem) returns a
solution or failure
inputs  : problem, a problem
for depth ← 0 to ∞ do
result ← Depth-Limited-Search (problem, depth)
if result ≠ cutoff then return result
```

The iterative deepening search algorithm, which repeatedly applies depth-limited search with increasing limits.

It terminates when a solution is found or if the depth- limit search returns **failure** meaning that no solution exists.

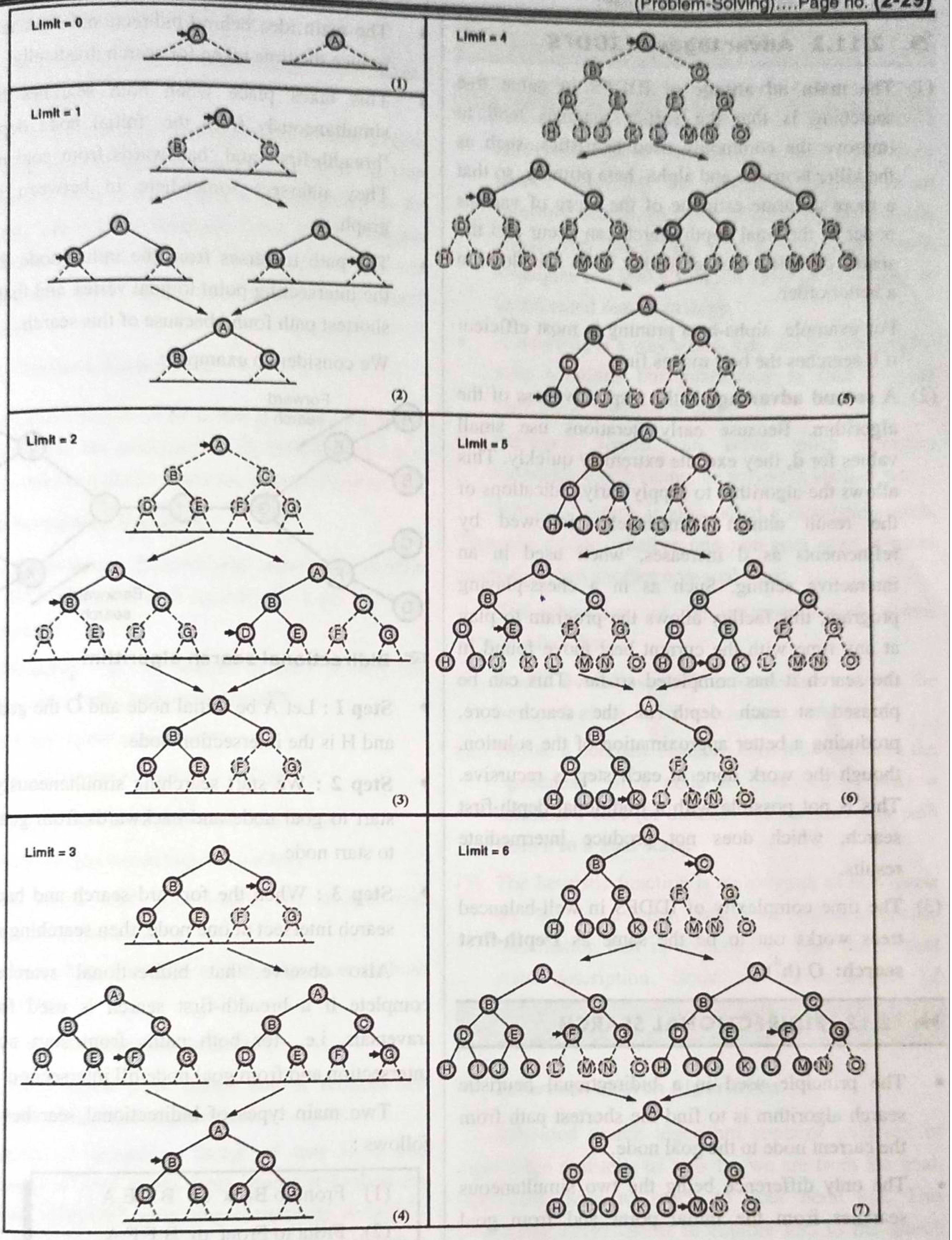

Fig. 2.11.1

Scanned by TapScanner