Lecture 6 (More Search Techniques)

1 Improved DFS (Backtracking)

1. Only one successor is generated at a time rather than all successors
2. Memory is saved by *modifying* current state instead of copying

2 Depth-Limited Search

Fix the maximum goal depth and terminate DFS to the max permitted depth (l)

1. Time complexity: $O(b^l)$
2. Space complexity: $O(bl)$
3. Incomplete (when $d > l$)
4. Sub-optimal (when $l > d$)

3 Iterative Deepening Search

```
function IterativeDeepeningSearch(problem):
    for depth = 0 to inf:
        result = DepthLimitedSearch(problem, depth)
        if result != failure:
            return result
```

1. Time complexity: $O(b^d)$
2. Space complexity: $O(bd)$
3. It is complete
4. Optimal solution is obtained

4 Bi-Directional Search

1. Spread out from both the start and goal states, find intersection
2. Space and time complexity: of the kind $O(b^{d/2})$
3. Needs an efficient `Predecessor` function (for the path from goal state)

4. To handle the case of multiple goal states, a dummy goal state is considered whose predecessors are goal states

5 Uniform Cost Search

1. Expand the *cheapest* node first
2. Frontier is a priority queue