LAB-2

REPORT

CS-366: ARTIFICIAL INTELLIGENCE

SUBMITTED BY:
Abhijeet Kumar (2101CS02)
Naitik Raj (2101CS49)
Satyam Kumar (2101CS72)

Que 1.

Comparing number of steps taken by UCS and IDS:

Example - 1:

// 0 here represents BLANK state
Randomly Generated Starting State = [[3,2,1], [4,5,6], [8,7,0]]

Target State = [[1, 2, 3], [4, 5, 6], [7, 8, 0]]

Algorithm	**Number of steps**	**Status**
UCS	25	Reachable
Iterative Deepening	25	Reachable

Que 2.

Working of Algorithms:

1. Uniform Cost Search: UCS is a searching algorithm for graph with different edge weights.

   Here's how UCS works:
   Initialisation: We start from the initial state. Push the initial state in the open list.
   Expansion: From the open list, we choose the state at shortest distance and generate all it's neighbours, and distance to reach them. If any generated node is not in the open list, we add it to the list.
   Goal Check: We check if the current state in the target state If yes, we have found the optimal path, else, continue.
   Update list: Pop the state that has been explored from the open list. **Repeat**: We repeat the above steps until either we find the solution or traverse the maximum number of steps possible.

2. Iterative Depth Search: is a search algorithm that combines the advantages of Breadth-First Search (BFS) and Depth-First Search (DFS).

   Here's how IDS works:

   Initialisation: We start at the initial state. Determine the maximum search depth.
   DFS Search: We apply DFS with the current depth limit.
   Goal Check: We check the current state for target state. If yes, we have found the optimal path.
   Increase search depth limit: If we couldn't find the target state while searching in the previous iteration, we increase the depth limit.
   Repeat: We repeat the process until either we find the state or exhaust the depth limit.

Comparison based on optimality:
The choice of optimal algorithm depends on the problem and varies from one situation to another. The search algorithms (BFS, DFS, UCS and IDS) perform differently under differnt circumstances, thus have their own advantage and disadvantages:

Breadth First Search is optimal for finding the shortest path in an unweighted graph.
The algorithm explores all nodes at a given depth level before moving deeper.

- BFS guarentees optimal path.
- BFS requires a lot of memory to store the visited states.

Depth First Search is optimal for finding solutions that are far from the initial state.

- DFS doesn't guarentees optimality, as it might search along a longer branch first before backtracking and exploring a shorter path. - DFS requires less memory (than BFS).

The **Iterative Deepening stratergy** is a combination of BFS and DFS. It predefines the search depth (which it increases gradually) and searches in a DFS-like manner. It makes use of completeness of BFS and uses much less storage.

- It guarentees optimal solution.
- It is a memory-efficient algorithm.

Uniform Cost Search also functions like BFS, but, unlike BFS it has the ability to identify a optimal path later in its execution. It is optimal for searching in graphs with vatied edge weights.

- It also guarentees optimal path.
- Similar memory requirements as BFS.
