Vedant Pathare
9564, Comps-A
Batch B
AI

* Experiment-5: (Postlab)

Q.1) Explain the time complexity of the A* algorithm.

Ans:- The time complexity of A* algorithm depends on several factors like branching factor, depth of solution and optimality of heuristic function. The time complexity of A* can be expressed as $O(b^d)$ in the worst case. However, the actual time complexity is often much lower in practice due to the effectiveness of the heuristic function in guiding the search towards the goal.

Q.2) What are the limitations of A* algorithm?

Ans:- i) Admissible Heuristic: A* requires an admissible heuristic to guarantee optimality. If the heuristic is not admissible, A* may return a suboptimal solution.
ii) Memory Usage:- A* may require significant memory to store the explored nodes, especially in large search spaces with a high branching factor & depth.
iii) Informedness:- A* heavily relies on the quality of the heuristic function. If the heuristic is not informative enough or leads the search in the wrong direction, A* may perform poorly.

Q3) Discuss A*, BFS, DFS and Dijkstra's algorithm in detail with examples.

Ans:- A* Algorithm:- It is an informed search algorithm that combines the advantages of both uniform cost search and heuristic search. It guarantees optimality when using an admissible and consistent heuristic. A* efficiently explores the most promising paths first based on the estimated cost to reach the goal. Eg:- A* would use a heuristic function to guide the search towards the goal node efficiently while ensuring optimality if the heuristic is admissible.

BFS:- It is an uninformed search algorithm that explores all neighbours of a node before moving on the next level of the search tree. BFS guarantees finding the shortest path in unweighted graphs but may require significant memory in graphs with a large branching factor.

DFS:- DFS is an uninformed search algorithm that explores as far as possible along each branch before backtracking. DFS does not guarantee optimality and may get stuck in infinite loops if cycles exist in the graph. DFS is often used in problems where finding any solution is sufficient, rather than the optimal solution.

Dijkstra's Algorithm:- It is a uniform cost search algorithm used to find the shortest path in weighted graphs with non-negative edge weights. Dijkstra's algorithm guarantees optimality when all edges are non-negative. It explores nodes in increasing order of their shortest path distances from the source node.