Girish Nhavkar
9560 TE Comps A

Page No.
Date

AI Exp-5

Postlab

Q.1] T.C of A* algo

→ T.C of A* depends on quality of heuristic func^n. In worst-case, the algo can be O(b^d) where b = branching factor the avg no of edges from each node

Q.2] What limitations of A* algo?

→ A* algo, while renowned for efficiency, has limitations

1.] computational cost

→ A* can be computationally expensive, especial in scenarios: Extensive search space: when dealing with vast no of paths, the exploration process can become resource-intensive

High branching factors: If each node in search space has potential, the algo needs to evaluate numerous options

2] Reliance on heuristics:

A* heavily relies on quality of heuristic func used to estimate dist to goal

Poor heuristic: A poor design can lead algo down suboptimal paths.

Domain-specific: Design an effective heuristic often req significant domain

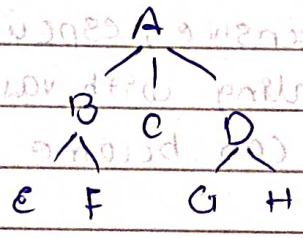
Q. 8] Discuss A*, BFS, DFS & Dijkstra algo?

A] BFS)

BFS is an algo for traversing tree or graph D.S
It starts at root, explores all of neighbor nodes at present depth prior to moving on nodes at next depth level

- BFS uses queue D.S to keep track of next nodes
- BFS guarantee shortest path in terms of numbers

Eg



If we perform BFS starting from node A

A → B → C → D → E → F → G → H

DFS)

- Graph D.S
- Starts at root explores as far as possible before backtracking
- uses stack D.S to keep track of next node
- DFS does not guarantee shortest path. It might traverse deep into branch

Eg

same graph

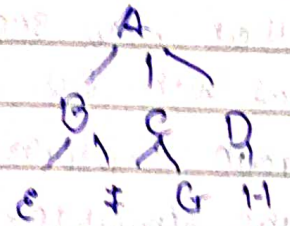
A → B → E → F → C → D → G → H

Page No.
Date

Dijkstra's algo

- used to find shortest path from single source vertex to all other vertices in graph
- maintains shortest distance from source vertex
- It iteratively selects vertex with smallest dist from source & updates dist

Eg



A* algo :

- To find short path in graph btw start node & target node
- uses hruristic to estimate cost from current node to goal
- A* keep track of total estimated cost of reaching node called f-score, which is sum of cost from start to current node