BSCS 23177

AI Assignment-01

PART A

Question 1.

A* search is a widely used algorithm for finding the shortest path in a graph or tree, and it ensures an optimal solution under specific conditions related to its ~~ret~~ heuristic function Here's how it works and conditions required for optimality:

A* search evaluates nodes by combining two costs:

- $g(n)$: The cost ~~of~~ to reach the node n from the start node.
- $h(n)$: The estimated cost from node n to the goal

The total cost function is given by:

$$f(n) = g(n) + h(n)$$

Admissible Heuristic: A heuristic $h(n)$ is admissible if it never overestimates the cost to reach the goal. This means:

$h(n) \leq$ true cost from n to the goal

Admissibility ensures that $f(n)$ never overestimates the true cost of a solution through n.

Consistent Heuristic: A heuristic $h(n)$ is consistent if, for every node n and every successor n' of n the estimated cost of reaching the goal from n is no greater than the step cost of getting to n' plus the estimated cost of reaching the goal from n':

$$h(n) \leq c(n,a,n') + h(n')$$

This is a form of the triangle inequality and ensures that the heuristic is locally consistent

Proof of Optimality

Tree Search: A* is optimal if $h(n)$ is admissible. This is because $f(n)$ will never overestimate the true cost, and A* will always expend the node with the lowest $f(n)$, ensuring the optimal path is found

Graph Search: A* is optimal if $h(n)$ is consistent. Consistency ~~ensure~~ ensures that the values of $f(n)$ along any path are nondecreasing, which means A* will always expand nodes in nondecreasing order of $f(n)$. This ~~garant~~ guarantees that the first goal node selected for expansion is part of an optimal solution

A* might fail to find an optimal solution in the following scenarios:

- Non-consistent Heuristic: if the heuristic is admissible but not consistent, A* might not expend nodes in the correct order, potentially leading to a suboptimal path.

Graph search without proper handling: In graph search, if nodes are revisited without proper handling (eg not updating the path cost correctly), A* might not find the optimal path.

Example of Sub optimal path: consider a weighted graph where the heuristic $h(n)$ is admissible but not consistent for instance, ~~suppose~~ suppose we have a graph with nodes A, B, and C where:

A is the start node

C is the goal node

The true costs ~~A~~ are $A \rightarrow B = 2$, $B \rightarrow C = 3$, $A \rightarrow C = 6$

The heuristic values are $h(A) = 5$, $h(B) = 1$, $h(C) = 0$

Here $h(A)$ is admissible because it does not overestimate the cost to reach C. However it is not consistent because $h(A) \leq c(A,B) + h(B)$ does not hold $(5 \leq 2+1)$ A* might choose the path $A \rightarrow C$ with a cost of 6 instead of the optimal path $A \rightarrow B \rightarrow C$ with cost of 5

Question 2

DFS (depth first search)

Advantages

- memory efficient: DFS uses less memory compared to breadth first search (BFS) because it only needs to store a path from the root to a leaf node.

Complete: DFS is complete if the branching factor is finite, meaning it will eventually find a solution if one exists

Limitations:

- Infinite Paths: DFS can get stuck in an infinite path in graphs with cycles or infinte

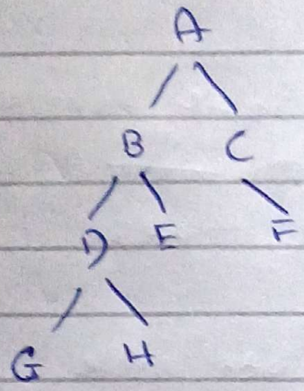
depth, leading to non-termination
Non-optimal Solutions: DFS may find a solution but it is not guaranteed to be the optimal (shortest) one.

Depth limited search (DLS)
Advantages
Avoids Infinite paths: by setting a depth limit, DLS avoids the problem of infinite paths, making it more practical for certain types of graphs
Memory Efficient: like DFS, DLS is memory efficient as it only needs to store nodes up to the depth limit
Limitations:
Incompleteness: DLS is not complete because it may miss solutions that lie beyond the depth limit
Non-optimal solutions: similar to DFS, DLS does not gurantee finding the optimal solution

Scenarios to prefer DLS over DFS
1. In graphs with cycles, DFS can get stuck in an infinite loop. DLS can avoid this by limiting the search depth, ensuring termination
2. When a computational resources are limited, and a quick, though potentially suboptimal, solution is needed, DLS can be more efficient than DFS

Example Tree

DFS and DLS iteration on tree with depth limit of 3



DFS iteration:

A -> B -> D -> G -> H -> E -> C -> F

DLS iteration: (limit 3)

A -> B -> D -> C -> E -> F

Question 3

statement 1:

DFS always expends at least as many nodes as A* search with an admissible heuristic

false, DFS does not necessarily expand at least as many nodes as A* search with an Admissible heuristic. DFS explores as deep as possible along each branch before backtracking which can lead to exploring many nodes that are not part of the optimal path. In contrast A* search with an admissible heuristic focuses on expending nodes that are likely to be part of the optimal path ,potentially expanding fewer nodes than DFS

statement 2:

Manhattan distance ~~m~~ is an admissible heuristic for the problem of moving the rook from ~~square~~ square A to square B in the smallest number of moves.

True. The Manhattan distance is an admissible heuristic for this problem. The Manhattan distance between two points on a grid is the sum of the absolute differences of their coordinates. For a rook on a ~~the~~ chessboard, which can move vertically or horizontally, ~~th~~ ~~than~~ the Manhattan distance represents the minimum number of ~~not~~ moves required to reach from one square to another without any obstacles. Since it never overestimates the cost (number of moves), it is admissible.

PART B

Question 2

1) The missionaries and cannibals problem can be formulated as a state-space search problem.
A state representation by a tuple (M, C, B) where
- M is number of missionaries on the original side of the river
- C is the number of cannibals on the original side of the river
- B indicates whether the boats is on the original

side (1) or the other side (0)

Initial state:

(3, 3, 1) all missionaries and cannibals are on the the original side and the boat is there too

Goal state

(0, 0, 0) All missionaries and cannibals are on the other side, and the boat is there too

Valid moves

the boat can carry 1 or 2 people across the river valid moves must ensure that missionaries are never outnumbered by cannibals on either side of the river

Constraints:

$0 \leq M \leq 3$

$0 \leq C \leq 3$

boat position B is binary (0 or 1)

missionaries cannot be outnumbered by cannibals on either side