Artificial Intelligence - Assignment 01

PART - A (Conceptual Questions)

1. Define admissible heuristic. Give one example of an admissible heuristic and one example of a non-admissible heuristic in a grid-world maze.

→ An admissible heuristic is a guess about the cost to the goal that is always optimistic. It never guesses a higher cost than the real one.

- Admissible Example: The Manhattan Distance in a grid maze (counting squares over and down). It is the best-case path if there were no walls, so it never overestimates.

- Non-Admissible Example: Using the Manhattan Distance multiplied by 3. This guess would be too high and would mislead the search algorithm.

2. Define consistent heuristic. Explain how it is related to the triangle inequality.

→ A consistent heuristic means that as you move to a neighboring spot, your estimated cost to the goal doesn't drop by more than the cost of that single step.

This is like the triangle inequality: taking a step to a neighbor and then continuing to the goal should never seem shorter than your original estimate. It ensures the estimates are sensible from one step to the next.

3. Can a heuristic be admissible but not consistent? Provide reasoning.

→ Yes, a heuristic can be admissible (optimistic overall) but not consistent (making weird jumps between neighbors).

Reasoning: Imagine moving from A to B costs 1, but your heuristic estimate drops from 5 down to 0. This breaks the consistency rule because the estimate dropped by much more than the step cost. However, as long as the actual path from A to the goal was 5 or more, the heuristic was still admissible (it didn't overestimate).

4. Why is admissibility necessary for A^* to be optimal? Why is consistency necessary?

→ • Admissibility is for Optimality: It ensures A^* finds the best path. If your heuristic overestimates the cost, the algorithm might ignore the actual best route because it looks too expensive at first glance.

• Consistency is for Efficiency: It makes A^* faster. It guarantees that once the algorithm explores a spot, it never has to backtrack to check it again. This avoids re-doing work and speeds everything up.

PART - B (Worked Examples)

1. Consider the following graph :

A -- 2 --> B -- 2 --> C (goal)
A -- 5 --> C (goal)

Case (i) - With h (A) = 3 , h (B) = 1 , h (C) = 0 , show that A^* finds the (optimal) path.

Case (ii) - With h(A) = 5 , h(B) = 5 , h(C) = 0 , show step-by-step how A^* fails to find the optimal path.

→ Lets trace A^* on the given graph to see how the heuristic affects the outcome. The optimal path is A → B → C with a total cost of 4.

Graph :

- A -- 2 --> B
- B -- 2 --> C (Goal)
- A -- 5 --> C (Goal)

Case 1 : h (A) = 3 , h (B) = 1 , h (C) = 0 (Admissible Heuristic)

This heuristic is admissible because h (A) = 3 is less than the true cos of 4, and h (B) = 1 is less than the true cost of 2.

- **Step 1 :** Start at A. Evaluate paths to its neighbors, B and C.
  - → Path $A \to B$: Cost = $g(B) + h(B) = 2 + 1 = 3$
  - → Path $A \to C$: Cost = $g(C) + h(C) = 5 + 0 = 5$

- **Step 2 :** A^* chooses the path with the lowest score, which is the path to B (score 3).

- **Step 3 :** From B, we can only go to C. Let's calculate the cost of the full path $A \to B \to C$.
  - → Path $A \to B \to C$: Cost = $g(C) + h(C) = (2+2) + 0 = 4$

- **Step 4 :** The algorithm now compares the new path to C (score 4) with the direct path to C (score 5). It sees that 4 is better.

- **Result :** A^* expands the path through B and finds the goal. The final path is $A \to B \to C$ with a cost of 4. This is the optimal path.

Case 2 : $h(A) = 5$, $h(B) = 5$, $h(C) = 0$ (Non-Admissible Heuristics)

This heuristic is non-admissible because $h(A) = 5$ overestimates the true cost of 4, and $h(B) = 5$ overestimates the true cost of 2.

• Step 1 : Start at A. Evaluate paths to its neighbors, B and C.
Path $A \to B$: Cost = $g(B) + h(B) = 2 + 5 = 7$
Path $A \to C$: Cost = $g(C) + h(C) = 5 + 0 = 5$

• Step 2 : A* chooses the path with the lowest score. This time, it is the direct path to C (score 5), because the path through B looks too expensive (score 7).

• Result : A* immediately finds the goal via the path $A \to C$ with a cost of 5. It stops without ever exploring the path through B. This is not the optimal path. The bad heuristic misled the algorithm.

2. For the same graph, test consistency :
Case (i) : $h(A) = 3$, $h(B) = 1$, $h(C) = 0$.
Case (ii) : $h(A) = 4$, $h(B) = 5$, $h(C) = 0$.
Identify which violates the consistency condition.

→ The rule of consistency is : $h(\text{start}) \leq \text{cost}(\text{start}, \text{end}) + h(\text{end})$

- <u>Case 1</u> : $h(A) = 3$, $h(B) = 1$, $h(C) = 0$

  - Edge $A \to B$: Is $h(A) \leq cost(A,B) + h(B)$?
    $\Rightarrow 3 \leq 2+1$? Yes, $3 \leq 3$. (Consistent)

  - Edge $B \to C$: Is $h(B) \leq cost(B,C) + h(C)$?
    $\Rightarrow 1 \leq 2+0$? Yes, $1 \leq 2$. (Consistent)

  - Edge $A \to C$: Is $h(A) \leq cost(A,C) + h(C)$?
    $\Rightarrow 3 \leq 5+0$? Yes, $3 \leq 5$. (Consistent)

Conclusion : This heuristic is <u>consistent</u>.

- <u>Case 2</u> : $h(A) = 4$, $h(B) = 5$, $h(C) = 0$

  - Edge $A \to B$: Is $h(A) \leq cost(A,B) + h(B)$?
    $\Rightarrow 4 \leq 2+5$? Yes, $4 \leq 7$. (Consistent so far)

  - Edge $B \to C$: Is $h(B) \leq cost(B,C) + h(C)$?
    $\Rightarrow 5 \leq 2+0$? NO! $5 \nleq 2$. (Inconsistent)

Conclusion : This heuristic violates the consistency condition on the path from B to C.

PART - D (Critical Thinking)

1. Why A* is guaranteed optimal if both admissibility and consistency hold?

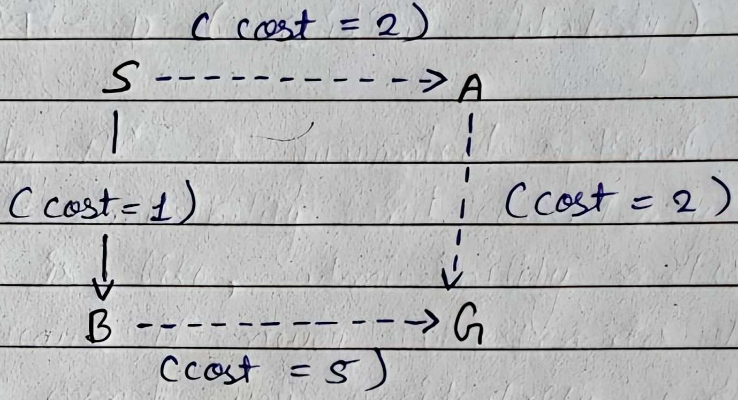
⇒ It is like, admissibility makes A* an optmist. Because it never overestimates the real cost, it won't ever accidentally ignore the true best path. It keeps all promising options on the table.

Consistency, on the other hand, makes A* efficient. It ensures cost estimates are sensible between neighbors, so once A* explores a node, it knows it found the cheapest way there. This prevents the algorithm from wasting time by second, guessing and revisiting nodes.

Together admissibility finds the best path, and consistency finds it without wasted effort.

2. Maze example where admissibility fails and show A* produces a non-optimal result.

⇒ Consider a maze where the optimal path (S → A → G) costs 4 and a suboptimal path (S → B → G) costs 6.



The Heuristic (Non - Admissible):

Let's use a heuristic $h(n)$ that overestimates the cost from A.

- $h(A) = 10$ (Non-admissible, true cost = 2)
- $h(B) = 4$ (Admissible, true cost is 5)

How A Fails :

A* calculates $f(n) = g(n) + h(n)$ to decide which node to explore.

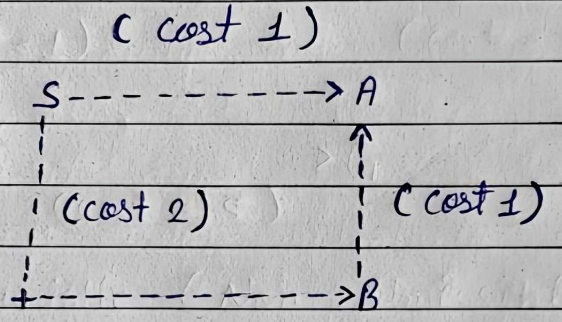
- Path via A : $f(A) = 2$ (cost to A) + 10 (h) = 12
- via B : $f(B) = 1$ (cost to B) + 4 (h) = 5

Because, $f(B)$ is lower.

A* is deceived into exploring the path through B first. It finds the goal with a cost of 6 and terminates, incorrectly reporting this as the best path without ever exploring the truely optimal route through A.

3. Example of consistency fail and show how A* may expand nodes incorrectly.

→



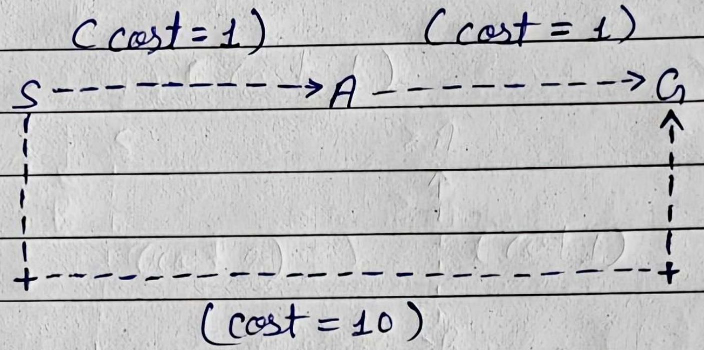
S → A (cost of 1)
S → B → A (cost of 2+1 = 3)

Inconsistent Heuristic :
$h(S) = 3$, $h(B) = 0$, and $h(a) = 5$. This is inconsistent because $h(S) > \text{cost}(S, B) + h(B)$ (since $3 > 2+0$).

A* explores B first because it looks cheaper. It then finds a path to a through B. Later, when it evaluates the direct S → A path, it realises it found a better way to get to A and has to re-evaluate it. This wastes time.

4. Example which is admissible but inconsistent.

→



The best path is $S \rightarrow A \rightarrow G$ (cost = 2).

The Heuristic :

- $h(S) = 1.5$ (Admissible, since $1.5 \leq 2$)
- $h(A) = 0$ (Admissible, since $0 \leq 1$)

Why it is inconsistent :

S to A : $h(S) \leq \text{cost}(S, A) + h(A)$.

- Is $1.5 \leq 1 + 0$?

No! $1.5 \nleq 1$. It is inconsistent.

The estimated cost drops from 1.5 to 0 after taking a step that only cost 1. This share drop violates consistency, even though both guesses were optimistic (admissible).

Conclusion:

What I learned about admissibility and consistency in A* search is that they are the two key rules that make the algorithm both smart and fast. Admissibility is the "optimism" rule: by never overestimating the cost, it guarantees A* will find the actual shortest path. Consistency is the "efficiency" rule: it ensures the guesses are steady and realistic, which prevents A* from wasting time re-checking the same spots.
In short, admissibility guarantees the best solution, while consistency makes sure the search runs smoothly.