# Assignment - 01

Zarin Tasnim Haider

ID 21301380

Section 08

CSE422

## Question 1

A* Algorithm :

```
function AStarSearch (initial):
    open = {initial}
    closed = {}

while open is not empty
    current = node in open with lowest f
    remone current from open add current to closed

    if goaltest(current):
        return reconstruct(current)

    for each neighbor in neibors(current):
    if neighbor is in closed:
        contine

    gScore = g(current) + cost(current, neighbor)

    if neighbor not in open or gScore < g(neighbor):
        set g(neighbor) to gScore.
        set h(neighbor) to h(neighbor)
        set f(neighbor) to g(neighbor) + h(neighbor)
        if neighbor not in open:
            add neighbor to open.

    return failure
function reconstruct (node):
    path = []
    while node is not null:
        add node to path
        node = node.parent
    return reversed(path)
```

## Ans to the Q.No-2

The goaltest(n) function in search algorithms plays a crucial role in determing whether a given node n satisfies the goal conditions. Here are two problem scenarios illustrating its importance.

1) **Pathfinding in a Maze:**

**Problem Description:** Consider a scenario where a robot needs to navigate through a maze from the start point to the goal.

**Importance of goaltest(n):**

The goaltest(n) function would check if the current node represents the robot's position at the goal location. Without the function the search algorithm might continue exploring paths indefinitely, leading to inefficiently and potentially not reaching the goal.

2) **Puzzle solving - Eight Puzzle Game**

**Problem Description:** Imagine an eight puzzle game where tiles are arranged in a 3x3 grid, and the objective is to reach a specific configuration by sliding tiles into empty spaces.

## Importance of goalTest(n)

In this context the goalTest(n) function ensures that the current node represents the puzzles configuration that matches the desired solution. Without it, the search algorithm might explore various configurations without identifying when the puzzle has been successfully solved.

## Answer to the Q. 3

Start = Arad.
Step = Bucharest

Arad $^{0+366=366}$

Arad $^{366}$ | Zerind $^{0+75+374=449}$ | Sibiu $^{0+140+253=393}$ | Timi $^{8+118+329=497}$

Sibiu $^{393}$ | Zerind $^{449}$ | Timi $^{447}$ | Rimini $^{0+140+80+193=413}$ | Fagras $^{0+140+99+176=415}$

Rimni $^{413}$ | Zerind $^{449}$ | Timi $^{447}$ | fagras $^{415}$ | Pitesi $^{317+100=417}$ | Craiova $^{366+160=526}$

Fagras $^{415}$ | Zerind $^{449}$ | Timi $^{447}$ | Pitesi $^{417}$ | Craiova $^{526}$ | Bucharest $^{450+0=450}$

Pitesi $^{417}$ | Zerind $^{449}$ | Timi $^{447}$ | Craiva $^{526}$ | Bucharest $^{450}$ | Bucharest $^{418+0=418}$

Bucharest $^{418}$ | Zerind $^{449}$ | Timi $^{447}$ | Craiova $^{526}$ | Bucharest $^{450}$ * neighbors

Optimal Path: Arad → Sibiu → Rimnicu Vilcea → Pitesi → Bucharest.

Total Cost : 418 Km

## Ans to the Q. NO – 4

In A* Algo If the node (n) is not the goal then then childrens are pushed into the frontier. The goalTest (n) happens before the childrens are being pushed in the frontier. But if we do the goal test (n) on children before pushing into the frontier it would not provide us with an optional solution.

If we see the simulation:

when we pop Fagora!

Fagoras 415 | Zerind 449 Timi 447 Pitesi 417 Craiova 526 Bucharest 450

desired destination and children.

Here if we do goal test on the children of Fagoras which is Bucharest 450 then it will stop the algo and return us.

Arad → Sibiu → Fagoras → Bucharest.
Pathcost 450 (Sub → optimal solution)

But if we perform goal test on the node popped from the frontier (Fagaras 415) then the algo also would have gone on until it finds Bucharist 418. popping out of the frontier then it would have ben.
Arad → Sibiu → Rimnicuvilea → Pitesi → Bucharest.
Pathcost 918 (optimal solution)

So there is a optimitacity problem if we perform goal test on popped out nodes children

## Answere to the Q. NO- 5

Cosistency check : $h(n) \leq 6(n, a, c, h) + h(ch)$

Arad : $366 \leq 140 + 253 \Rightarrow 366 \leq 393$ [Sibiu]

$366 \leq 75 + 374 \Rightarrow 75 \leq 449$ [Zeriend]

$366 \leq 118 + 329 \Rightarrow 118 \leq 447$ [Timisoara]

Zeriend $374 \leq 71 + 380 \Rightarrow 374 \leq 451$ [Oradea]

Sibiu $253 \leq 151 + 380 \Rightarrow 253 \leq 531$ [Oradea]

$253 \leq 99 + 176 \Rightarrow 253 \leq 275$ [Fagaras]

$253 \leq 80 + 193 \Rightarrow 253 \leq 273$ [RimnicuVilcea]

Timisoara $329 \leq 111 + 244 \Rightarrow 329 \leq 355$ [Lugoj]

Lugoj $244 \leq 70 + 241 \Rightarrow 244 \leq 311$ [Mehadia]

Mehadia $241 \leq 75 + 242 \Rightarrow 241 \leq 317$ [Doberta]

Doberta $242 \leq 120 + 160 \Rightarrow 242 \leq 280$ [Craiova]

Rimki : $193 \leq 146 + 160 \Rightarrow 193 \leq 306$ [Craiova]

$193 \leq 97 + 100 \Rightarrow 193 \leq 197$ [Pitesi]

Craiova : $160 \leq 138 + 100 \Rightarrow 160 \leq 238$ [Pitesi]

Fagaras $176 \leq 211 + 0 \Rightarrow 176 \leq 211$ [Bucharest]

Pitesi $100 \leq 101 + 0 \Rightarrow 100 \leq 101$ [Bucharest]

Giurgiu: $77 \leq 90 + 10 \Rightarrow 77 \leq 90$ [Bucharest]

Neamt $234 \leq 87 + 226 \Rightarrow 234 \leq 313$ [Iasi]

Iasi $266 \leq 92 + 199 \Rightarrow 226 \leq 291$ [Vaslui]

Vaslui $199 \leq 142 + 80 \Rightarrow 199 \leq 222$ [Urziceni]

Urziceni $80 \leq 85 + 0 \Rightarrow 80 \leq 85$ [Bucharest]

$80 \leq 98 + 151 \Rightarrow 80 \leq 249$ [Hirsova]

Hirsova $151 \leq 86 + 161 \Rightarrow 151 \leq 247$ [Etoric]

~~For optimal path~~

~~consistancy : then~~

Here all nodes satisfy the $h(n)$ consistency, so the given heuristic in B.22 are consistant.

Ans to the Q.NO-6

Choosing the most dominant heuristic in the context of multiple heuristics for a problem is typically based on the idea of admissibility and informativeness.

1) Admissibility A heuristic is admissible if it never overestimates the tree ~~of~~ cost ~~of~~ to reach the goal from any given state. Admissible heuristics are important because they guarantee that the search algorithm, such as A*, will always find an optimal solutio

2) Informativeness:

An informative heuristic provides more accurat estimates of the remaining cost to reach the goal. In other words, it provides a closer approximation to the actual cost.

When multiple heuristics are available, choosing the most dominant one (the one that is more accurate or 'informative) ensures that the algorithm can make more 'informed decisions during the search process. This typically leads to a more efficient exploration of the state space and a faster convergence to the optimal solution.

If a less accurate heuristic dominates, it might lead to underestimation of the true cost, causing the algorithm to explore unnecessary paths or potentially miss the optimal solution.