# Theoretical Problems:

**Submission Deadline - 11:55PM, 9th August, 2017.**

**Name**: Manne Naga Nithin

**Roll**: \*\*\*\*\*\*\*\*\*

1. Describe in brief an autonomous vehicle as an intelligent agent. Give a **PEAS** specification for the task environment.
2. **An autonomous vehicle is essentially an intelligent agent, because it gets data from real time sensors in the car and it needs to act per a given set of rules, while driving towards its goal destination. As an intelligent agent, it needs to follow a set of constraints, while minimizing the risk and improving performance. It also needs to figure out the next safe state with minimal cost.**

**PEAS Specifications:**

**Performance Measures: Time taken to reach destination, Number of Accidents, Quick action to an event, Fuel Consumed, Number of people killed (should be 0).**

**Environment: People, Road condition, Signs, Signals, Climate, Vehicles**

**Actuators: Accelerator, Brake, Steering, Lights, Horn**

**Sensors: Camera, RADAR, Ultrasonic Sensors, Speed Indicator, GPS, Fuel Indicator**

1. Prove the following:
   1. A\* search is complete
   2. A\* search is optimal.
2. **A\* uses a heuristic function h(s), which is used to perform an approach like BFS, but using f(n) = h(n) + g(n), g(n)=minimum cost to reach ‘n’.**

**Since BFS is complete, A\* is also always complete.**

**With 2 nodes ‘a’ and ‘b’, ‘a’ being the predecessor of ‘b’,**

**f(b) = g(b) + h(b)**

**f(b) = g(a) + dist(a, b) + h(b)**

**with a consistent heuristic, dist(a, b)+ h(b) >= h(a)**

**f(b) >= g(a) + h(a)**

**f(b) >= f(a) => f is consistent.**

**This means that this algorithm is optimal.**

1. Weighted A\* can be described as best-first search with: f(n) = g(n) + w · h(n) or as f(n) = (1-w) · g(n) + w · h(n) . Using one of these formulations, provide a bound on the maximum sub-optimality of a path that weighted A\* can return as a function of w.
2. **f(n) = g(n) + w\*h(n)**

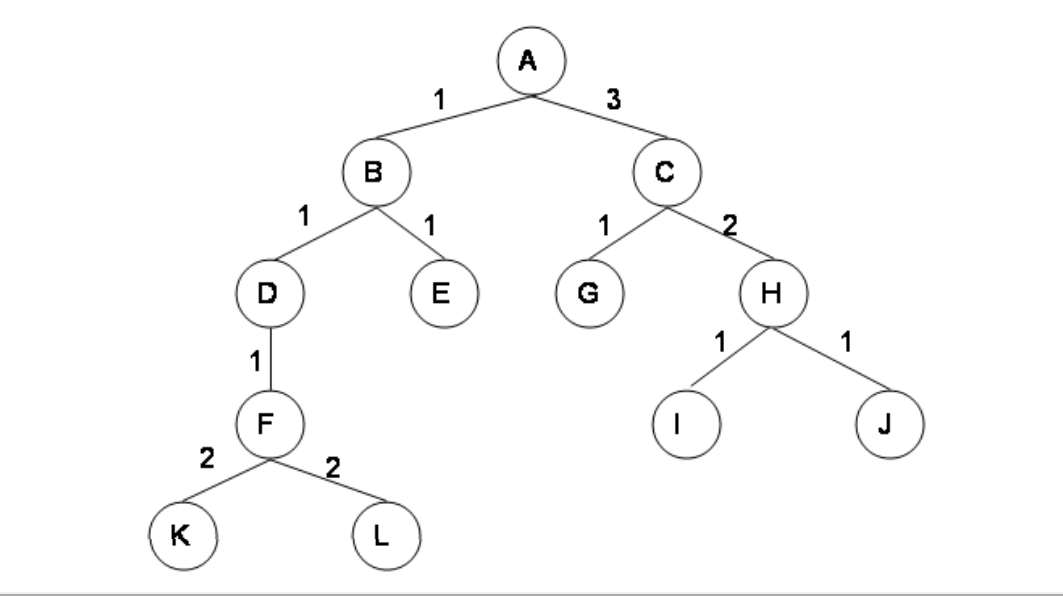
**if w=0, f(n)=g(n), equivalent to Dijkstra’s Algorithm**

**if w=1, f(n)=g(n)+h(n), A\* Algorithm**

**if w=inf, f(n) = h(n), Best first search**

**Therefore, optimality increases from w=0 to w=1, then decreases.**

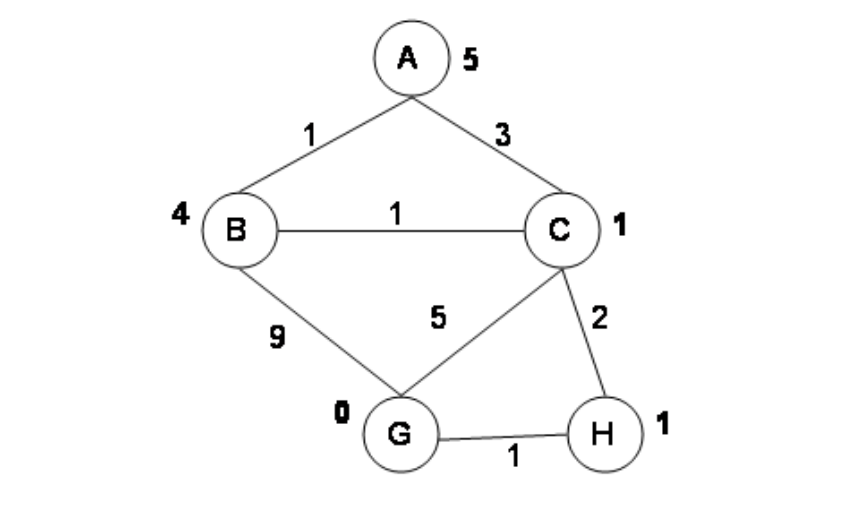
1. Consider the tree shown below. The numbers on the arcs are the arc lengths



Assume that the nodes are expanded in alphabetical order when no other order is specified by the search, and that the goal is state G. No visited or expanded lists are used. What order would the states be expanded by each type of search? Stop when you expand G. FIll the boxes with sequence of states expanded by each search.

|  |  |
| --- | --- |
| **Search Type** | **List of states** |
| Breadth First | A – B – C – D – E - G |
| Depth First | A – B – D – F – K – L – E – C - G |
| Iterative Deepening Search | [A] – [A – B – C] – [A – B – C – D – E – G] |
| Uniform Cost Search | A – B – D – E – C – F - G |

5. Consider the graph shown below where the numbers on the links are link costs and the numbers next to the states are heuristic estimates. Note that the arcs are undirected. Let A be the start state and G be the goal state.



Simulate A\* search with a strict expanded list on this graph. At each step, show the path to the state of the node that’s being expanded, the length of that path, the total estimated cost of the path (actual + heuristic), and the current value of the expanded list (as a list of states). Please transcribe (only) the information requested into the table given below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Path to State** | **Expanded Length of Path** | **Total Estimated Cost** | **Expanded List** |
| A | 0 | 5 | (A) |
| A - C | 3 | 3+1=4 | (C, A) |
| A - B | 1 | 1+4=5 | (B, C, A) |
| A - C - H | 5 | 3+2+1=6 | (H, B, C, A) |
| A - C - H - G | 6 | 3+2+1+0=6 | (G, H, B, C, A) |
|  |  |  |  |

6. Answer the following questions with respect to Problem 5.

1. Is the heuristic given in Problem 5 admissible? Explain.
2. **Yes, the heuristic is admissible, because for all nodes estimated cost is less than or equal to actual cost.**
3. **5<=5**
4. **4<=4**
5. **1<=3**

**H. 1<=1**

**G. 0=0**

1. Is the heuristic given in Problem 5 consistent? Explain.
2. **No, the heuristic is not admissible because for B, h(B)=4 is not less than or equal to [{dist(B, C)=1} + {h(C)=1}]=2.**
3. Did the A\* algorithm with strict expanded list find the optimal path in the previous example? If it did find the optimal path, explain why you would expect that. If it didn’t find the optimal path, explain why you would expect that and give a simple (specific) change of state values of the heuristic that would be sufficient to get the correct behavior.
4. **The above algorithm did not find the optimal path, which is A-B-C-H-G. This is because the heuristic is not consistent. It can be made consistent by changing h(C) to 3.**

7. The water jug problem can be stated as follows: you are given two jugs of capacities 4 litres and 3 litres. You also have a pump that can be used to fill either jug with water, and you can empty the contents of either jug at any time. Your goal is to get exactly 2 litres of water in the 4 litre jug.

1. Formulate this problem as a state-space search describing the state, move-generator and the goal checker.
2. **The state is (x, y), x being water in 4 l jug and y being water in the 3 l jug. Initial state is (0, 0), and the goal is (2, X), X can be anything. The moves can be:**
   * 1. **Fill the 4 l jug: (x, y) -> (4, y)**
     2. **Fill the 3 l jug: (x, y) -> (x, 3)**
     3. **Transfer to 4 l jug: (x, y) -> (4, x+y-4), if x+y>4, (x, y) -> (x+y, 0)**
     4. **Transfer to 3 l jug: (x, y) -> (x+y-3, 3), if x+y>3, (x, y) -> (0, x+y)**
     5. **Empty the 4 l jug: (x, y) -> (0, y)**
     6. **Empty the 3 l jug: (x, y) -> (x, 0)**
3. Suppose that it costs Rs. 5 every time the pump is used, Rs. 2 every time you fill the 4 litre jug and Rs 1 every time you fill the 3 litre jug. Find the lowest cost solution to this problem.
4. **The lowest path that I found is:**

**(0, 0) – (0, 3) – (3, 0) – (3, 3) – (4, 2) – (0, 2) – (2, 0)**

**Cost = (5+1) + (5+1) = 12**

**Assumption: Cost to fill 3 l jug = 5+1**

**Cost to transfer water between jugs = 0**

8. Decide whether the following statements are true or false. If true, explain why. If false, give a contradicting example. Recall that B is the average branching factor and L is the length of the shortest path from start to goal.

1. Bi-directional BFS is always faster than BFS when B ≥ 2 and L ≥ 4.
2. **False. Consider searching on a tree from the leaf to the root, and transitions always move towards the root. BFS is fast because branching factor is always 1, while BIBFS waste time on the half of the reverse part tracing a lot of branches. Note that time complexity is not the same as actual timing or number of nodes expanded. O() are worst case, and under a constant factor. On the other hand, in cases like A\* being more efficient than any optimal search given a heuristic, we really mean ALWAYS | it has been proved that for any search space no other optimal search can expand fewer nodes. Know the difference. Also, note that treating B as a constant is just an approximation, since the branching factor is different between nodes. Finally, you need to understand the assumptions underlying the analysis that BIBFS is faster than BFS.**
3. A\* search always expands fewer nodes than DFS does.
4. **False, A\* relies on a heuristic, which if not consistent can expand a lot more nodes than DFS will.**
5. For any search space, there is always an admissible and consistent A\* heuristic.
6. **True, h(s) = 0 is always admissible and consistent, but is equal to UCS**
7. IDA\* does not need a priority queue as in A\*, but can use the program stack in a recursive implementation as in DFS.
8. **True, because it works like DFS for every fmax value.**