Muhammad Umar Salman

21010241

ASSIGNMENT 2

Search and CSP's

1) Define
    a. State: A state is a situation in which an agent can find itself at a certain time period.
    b. State Space: The complete number of states that are possible from the initial state through sequence of actions of the agent.
    c. Search Tree: A sequence of actions and states that can be displayed originating from the initial state.
    d. Search Node: The current state after the action that led up to it.
    e. Goal: A desired state that the agent wishes to arrive at through its' actions which passes the goal test.
    f. Action: The agents influence on its' surroundings in an attempt to achieve or arrive at its' goal state.
    g. Transition Model: The description of what the actions do.
    h. Branching Factor: The maximum number of children that a given parent node could have.
2) Solve
    a. Search Strategies

        Note: BFS and DFS goal nodes are identified on generation, rest are identified on being dequed to the explored states.

        i. Depth First Search
            1. Explored = [A, B, E, F, C, G, H]
            2. Goal Reached = L
            3. Frontier = [K, L, D]
        ii. Breadth First Search
            1. Explored = [A, B, C, D]
            2. Goal Reached = I
            3. Frontier = [E, F, G, H, I, J]
        iii. Uniform Cost Search
            1. Explored = [A(0), B(5), C(6), D(7), E(7), F(8), H(9), I(10)]
            2. Goal Reached = I
            3. Frontier = [K(11), J(13), L(14), G(16)]
        iv. Greedy/Best Search
            1. Explored = [A(0), C(4) , H(0), L(0)]

2. Goal Reached = L
3. Frontier = [D(5), G(5), K(5), B(7)]

v. A* Search

1. Explored = [A(0), C(10), H(9), B(12), D(12), I(10)]
2. Goal Reached = I
3. Frontier = [L(14), F(14), E(15), K(16), G(21), L(22)]

b. The heuristic function h(n) is called admissible if h(n) is never larger than h*(n), namely h(n) is always less or equal to true cheapest cost from n to the goal where h(n) is the heuristic cost to the goal node and h*(n) is the actual cost to the goal node. Here since h(A) = 12 when the actual h*(A) = 10 we can see that for node A the heuristic is over estimating the cost. And the condition for admissibility is that for all states s h(s) <= h*(s). Hence this proves that h(N) is **not admissible.**

3) True or False

a. Depth-first search always expands at least as many nodes as A∗ search with an admissible heuristic.
False. Depth-first search may be able to expand fewer nodes than A*search with an admissible heuristic if through coincidence if DFS traverses directly to the goal node and one branch has a smaller g(n) + h(n) than the branch to the goal node is. In this case DFS would go directly to it whereas A* would have to backtrack from the smaller branch it took.

b. h(n) = 0 is an admissible heuristic for the 8-puzzle.
True. h(n)=0 can't over-estimates the remaining optimal distance to a goal node as 0 is not higher than any other number

c. A∗ is of no use in robotics because percepts, states, and actions are continuous.
False. Continuous state spaces can be abstracted to discrete states. All the percepts, actions and states change in robotics are based on pattern search. We do the search work first and then convert the results to continuous actions.

d. Breadth-first search is complete even if zero step costs are allowed.
True. BFS is optimal if all step costs are 1 or constant and is complete if d is finite. So, if a goal exists it will happen at finite depth d and regardless of the cost it will find the goal node making it complete and it finds it in O(bd) steps.

e. Assume that a rook can move on a chessboard any number of squares in a straight line, vertically or horizontally, but cannot jump over other pieces. Manhattan distance is an admissible heuristic for the problem of moving the rook from square A to square B in the smallest number of moves.

False. The Manhattan distance may over-estimate the optimal remaining number of moves to the goal because a rook may cover several squares in a single move. However, had the path cost instead been the number of squares covered, then Manhattan distance would be admissible.

4) Answer
   a. A constraint satisfaction problem (CSP) consists of:
      i. a set of variables {x1, x2, … xi}
      ii. a finite set of domain D possible values associated with each variable D1, D2, D3 . . . Dn where Di = {v1, v2 . . . vk}
      iii. a set of constraints C restricting the values that the variables can simultaneously take c1, c2, . . . c3.

      CSP is to assign values to variables so that all constraints are satisfied. Variables represent the entities in the constraint satisfaction problems and the domain of a variable is a set of possible values that can be assigned to the variables. The constraint are a set of relations upon domains of variables. A constraint on a set of variables restricts the values the variables can simultaneously take. It says what values are allowed and not allowed among the variables.

   b. Search trees are data structures to present the states of the search in systematic way. These techniques are carried out to traverse the search tree and find solutions. For CSPs, search tree can be constructed to include all possible (partial) assignments of values to variables. Nodes in a search tree represent (partial) solutions for the CSP whereas branches in a search tree represent possible assignments of values to variables.

   c. The solutions of CSP in a search tree will always be on the nth level. If we use the breath-first search, which searches all the nodes on the current level before it goes to the next level, the quickest it may find a solution will be after it finishes all the nodes on the 1st to the (n-1) th level. Using depth first search, there is a chance that the search may find a solution the first time it goes down to the nth level. Although this does not often happen, the chance of finding a solution quicker is still much higher than that of breath-first search.

Logic

1) Translate
   a. R ⟺ S
   b. S ⇒ R ∧ R ⇒ S which is also equal to S ⟺ R
   c. H ∧ HA
   d. C ∨ D

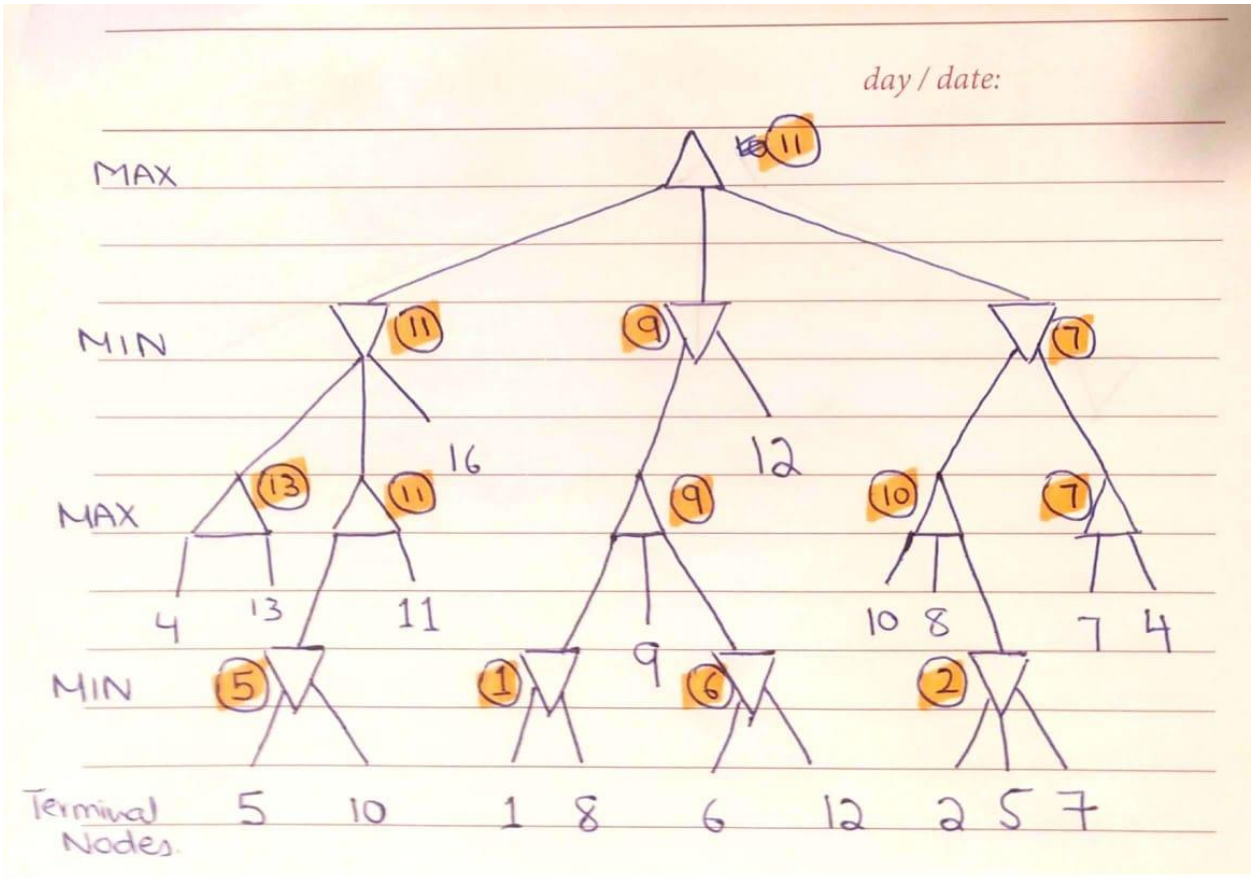
   Where ∧ is AND , ∨ is OR, ⇒ is implies, ⟺ is bidirectional

2) Sentence Validity

a. V
b. U
c. S
d. V
e. S
f. U

Game Playing

1) Solve



2) Solve