# 440/540 Artificial Intelligence HW1

| | |
|---|---|
| Last Name | Ghosh Dastidar |
| First Name | Priyanka |
| WSU ID | 11804812 |

# Q1. [20 pts] Intelligent Agents

Consider a system that provides online translation of telephone conversation between English and Japanese speakers.

**(a)** [10 pts] Discuss its performance measure, environment, actuators, and sensors (PEAS)

**Answer:**

Given, a system that provides online translation of telephone conversation between English and Japanese speakers. We need to evaluate its performance measure, environment , actuators, and sensors. A system to be performing in its optimum level, there should be an accuracy of translation between languages, should have proper network connectivity, user's location also plays an important factor. There should be a working translation software to convert speech in real-time and an audio input/output to allow users to communicate in real-time. Also, the network quality should be good to allow the system to perform ideally. To classify these factors into groups mentioned above:

- **Performance measure :**
    - **Translation Accuracy** : A system like online translation should accurately translate conversations from Japanese to English or from English to Japanese, whilst preserving the conversation's context and meaning, allowing it to serve its intended purpose of facilitating communication between languages. No system should interpret translation inaccurately. We have a wide range of software at our disposal for rapid translation requirements. For instance, after integrating AI for the machine translation of product listing names, eBay saw a 10.9% increase in their international trading. The BLEU score (Bilingual Evaluation Understudy), which is used by Google under AutoML to represent the model quality, measures how similar the candidate text is to the reference texts and values closer to one suggest texts that are more similar.
- **Environment :**
    - **Location of the User** : If we can't manage how data is transmitted to the client, we can't even regulate the speed at which it should arrive, which can affect connectivity over the network. It has been found that monitoring system performance and network bandwidth from several locations enables challenges to be narrowed down to the network level. We may therefore draw the conclusion that a user's location affects the system's performance because a difference in network speed and quality will affect the translation's quality.
    - **Connectivity of Network :** Another factor responsible for proper functioning of the system is the network connectivity. If the system operates in an online environment, then its performance is impacted by network connectivity.
- **Actuators :**
    - **Input/Output :** To allow the users communicating in real-time, audio input/output are required as working actuators to get the system performing
    - **Translation software :** The translation software does the task of converting speech from one language to another in real-time and execute the process of communication
- **Sensors :**
    - **Network quality :** Performance of the system is impacted by network quality so monitoring it is important and the network quality sensor takes care of that.
    - **Audio Input :** The audio input sensor records human voice and converts it to a digital format so that the translation software can process it.

**(b)** [10 pts] Is this environment (1) fully observable (Yes/No); (2) deterministic (Yes/No); (3) static (Yes/No); (4) discrete (Yes/No); (5) episodic (Yes/No)? Do provide reasons to explain your (Yes/No) choice.

**Answer:**

(1) The environment is not fully observable. The speaker's emotions, tone of voice, body language, etc., which can also carry crucial information and has the potential to alter the context of the conversation, are not accessible to the online translation system, which can only listen to and interpret the conversation's audio.

(2) The environment is not deterministic. The translation of the current conversation does not depend/ reference to previous conversation. The language is deterministic whether it's English or Japanese rest can't have clarity in it, hence not deterministic.

(3) The environment is not static. It is somewhat dynamic as there would be change of topic or dialect change and then the system must adjust its translation likewise

(4) The environment is discreet, yes. The system processes the quantized, sampled audio signals after a finite number of values have been created.

(5) The environment is episodic. System will treat each conversation as a different episode and will not retain information from previous conversations.

## Q2. [15 pts] Search Nodes

Consider the graph search framework we discussed in class on a search problem with max branching factor $b$. Each search node $n$ has a backward (cumulative) cost of $g(n)$, an admissible heuristic of $h(n)$, and a depth of $d(n)$. Let $n_c$ be a minimum-cost goal node, and let $n_s$ be a shallowest goal node.

For each of the following, give an expression that characterizes the set of nodes that are explored before the search terminates. For instance, if we asked for the set of nodes with positive heuristic value, you could say: for all $n$, such that $h(n) \geq 0$. Don't worry about ties (so you won't need to worry about $>$ versus $\geq$). If there are no nodes for which the expression is true, you must write "none."

**(a)** [5 pts] Give an inequality in terms of the functions defined above to describe the nodes $n$ that are explored in a breadth-first search before terminating.

**Answer :**

**Inequality: All n, such that:**

In a depth-first search, nodes are first explored by traveling the furthest down a path (set of nodes) before turning around and exploring a different path on the search tree. Therefore, BFS essentially navigates through nodes in their increasing order of depth and terminates when it reaches the minimum-goal node, meaning that throughout the process, all nodes will have the same or less depth than the minimum-cost goal. And the set of nodes explored before the search terminates would be : for all n, such that $d(n) <= d(n_c)$

**(b)** [5 pts] Give an inequality in terms of the functions defined above to describe the nodes $n$ that are explored in a best first search (or uniform cost search) before terminating.

**Answer :**

**Inequality: All n, such that** Using an evaluation function to determine which nearby node is the most promising, Best First Search aims to then explore that area. In best-first search , in other words uniform-cost search, nodes are navigated based on their cumulative cost which is g(n) , so the set of nodes traversed before the search terminates would potentially be : for all n, such that $g(n) <= g(n_c)$

**(c)** [5 pts] Give an inequality in terms of the functions defined above to describe the nodes *n* that are explored in an A* before terminating.

**Answer :**

**Inequality: All *n*, such that**: Informally, A* Search algorithms are said to have "brains" compared to other traversal methods. What it really means is that it is a clever algorithm, setting it apart from other traditional algorithms. The parts below go into more information about this fact. Therefore, nodes are explored using this method based on a combination of the cumulative cost g(n) and an acceptable heuristic h(n), which also occurs to estimate the cost of achieving the goal from n. Hence, the set of nodes explored before the search terminates would be for all n, such that $g(n) + h(n) <= g(n_c) + h(n_c)$

## Q3. [30 pts] Search in Graph

Consider the vacuum world problem with state space show in Figure 1 below. Let the state numbers assigned in Figure 1. Let the initial state be state 1 and the goal state be either 7 or 8.

Assume the order of in which states are added to frontier (in graph search) follow the order (S, R, L) in which actions are examined. Depending on whether we use breadth-first search (BFS) or depth-first search (DFS), the state at the front or back of the frontier will be expanded first.

**(a)** [10 pts] Given a trace of the BFS (graph-search) algorithm in the following style: show the search tree at each stage (repeated states are eliminated) with (1) nodes in **frontier** and **explored** annotated with different color; (2) indication of next node to be explored. Assume nodes are only tested for goal or put in **explored** when it is their turn to be explored.

**Answer:**

To start off with BFS(graph-search) algorithm, we have frontier (List of nodes to be explored) and explored (List of nodes that have been explored).

As per the algorithm , we need to initialize the frontier with state 1, then we check if it's the goal state or not. If so, then stop and return success. Then if not goal state, we explore its children in order (S,R,L). Then we continue the above steps until the goal state is reached.

In the diagram below, nodes marked in green are explored, blue ones are in frontier and yellow ones are expanded nodes.

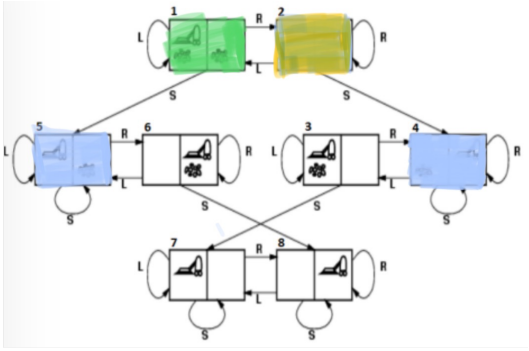
Below steps are followed for the algorithm :-

Adding Initial state (state 1) to the frontier

Examines 3 possible states (Suck,Left,Right)

Add resulting states (2 and 5) to the frontier

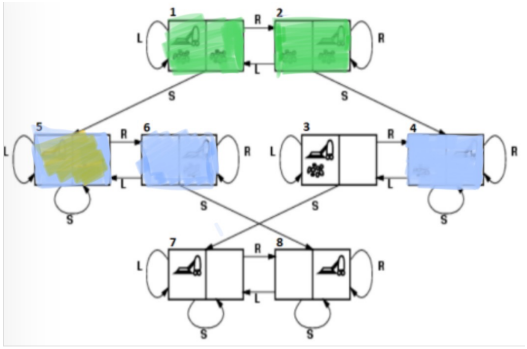
Mark state 1 as explored



State 2 will be expanded next

State 4 will be added to the frontier

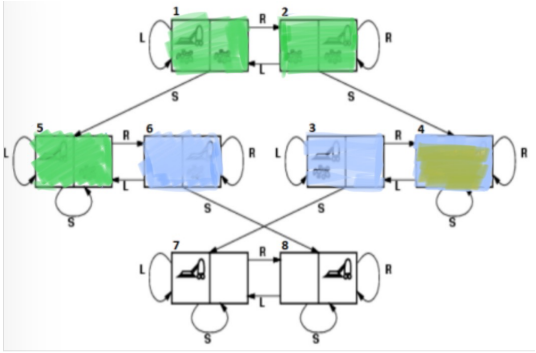
State 2 will be added to explored



State 5 will be expanded next

State 6 will be added to frontier

State 5 is added to  explored

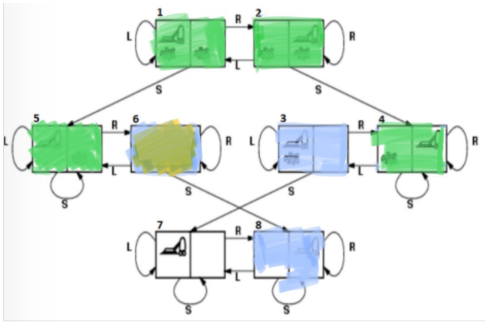


State 4 is expanded

State 3 is added to frontier

State 4 is added to explored

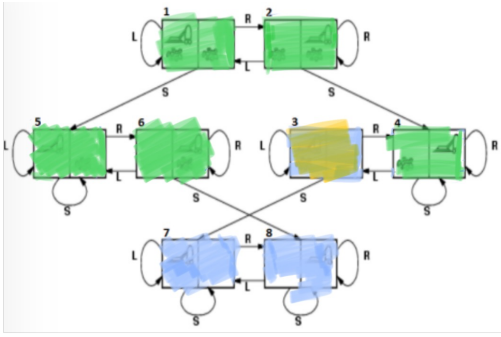
6



State 6 is expanded

State 8 is added to the frontier

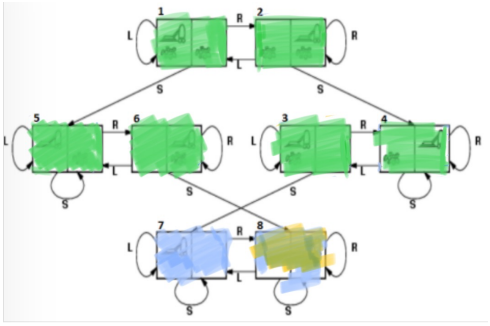
State 6 is added to explored



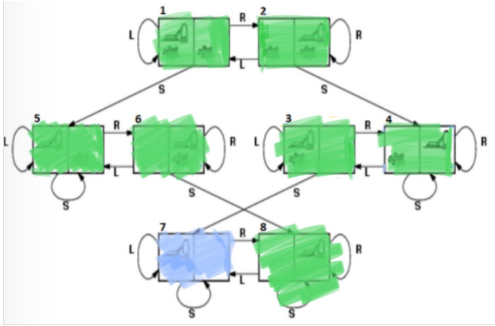
State 3 is expanded

State 7 is added to the frontier

State 3 is added to the explored



State 8 is expanded and added to the explored



Once state 8 is explored, state traversals are done as we have reached goal stage.

**(b)** [10 pts] Given a trace of the DFS (graph-search) algorithm in the following style: show the search tree at each stage (repeated states are eliminated) with (1) nodes in **frontier** and **explored** annotated with different color; (2) indication of next node to be explored. Assume nodes are only tested for goal or put in **explored** when it is their turn to be explored.

**Answer :**

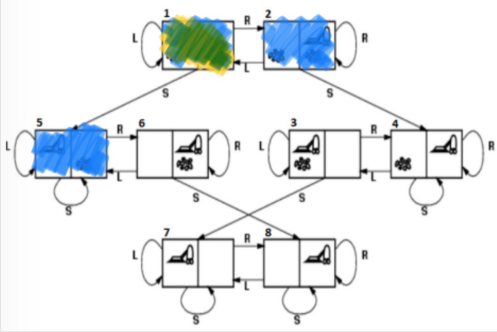
DFS algorithm starts at the root node and explore other nodes as far as possible before backtracking. Next node to be explored is taken from frontier and added to explored set. If we observe the node to be goal node, then search is successful, otherwise if the node has unvisited neighbor, we add it to frontier. The process continues until frontier is empty.

In the diagram below, nodes marked in green are explored, blue ones are in frontier and yellow ones are expanded nodes.

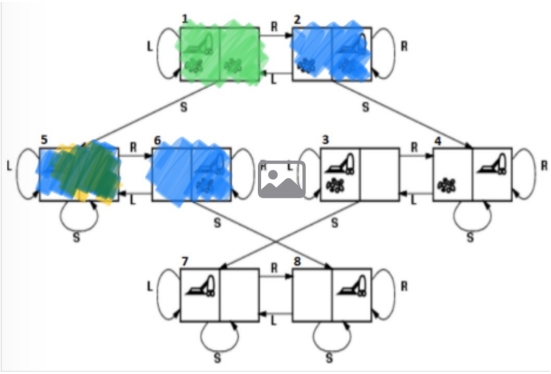
DFS starts from initial state 1, adding it to frontier.

State 1 is expanded

States 2 & 5 is added to the frontier.



State 1 is marked as explored.

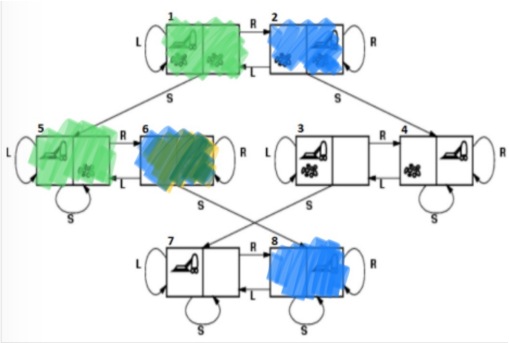


State 5 is expanded

State 6 is added to frontier

State 5 is marked as explored

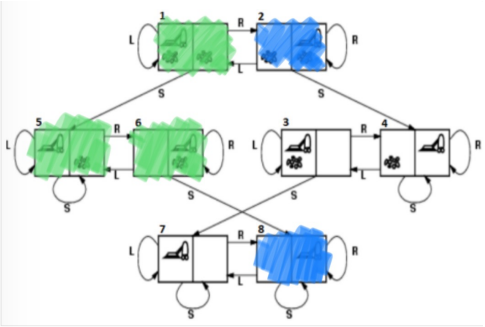
8



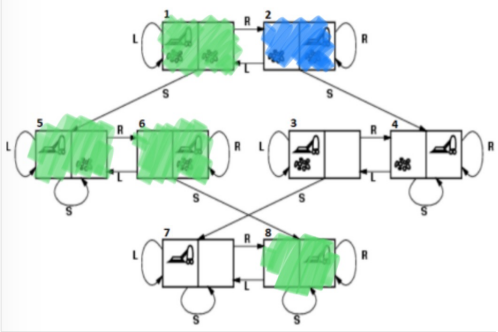
State 6 is expanded

State 8 is added to frontier

State 6 is explored



State 8 is expanded and marked as explored



**(c)** [10 pts] Which of these two algorithms is better for this problem? Is one search strategy always better than the other in general? Do provide explanation.

**Answer:**

The requirement for the problem will impact the choice of the algorithm. Breadth-first search will give us the optimal solution, one with the lowest cost. But BFS could be slow and might consume a lot of memory as it expands all the states at depth moving towards the deeper ones.

Comparatively, Depth-first search is much faster than BFS, given it expands a limited number of states before getting a solution. But DFS is not guaranteed to provide a optimal solution and may not be complete.

To conclude, no search strategy is better than the other. So, the choice of an algorithm depends on the trade-offs between memory, time and in finding an optimal solution. So, BFS could be a good choice when completeness and optimality are the most important factors, but DFS could be a good choice if speed and system's memory are important factors

# Q4. [20 pts] Adversarial Search

Consider the following game between two players: (1) there are two piles, each with two sticks; (2) each player might take either one or two sticks from an existing pile during his/her turn; (3) the player who picks the last stick wins.

**(a)** [5 pts] Propose a representation for the state of this game.

**Answer**:

Given, the problem statement, there are 4 sticks distributed over two piles and each player can pick a stick one or two sticks from the existing pile.

There could be possible two scenarios either player 1 winning or player 2 winning. We can represent the state of this game as a group of two integer , integers represent the number of sticks in each pile. When both piles have two sticks at the initial stage then it can be presented as (2,2).

If one player picks up a stick, then the state becomes (1,2). With each player's turn, the state gets changed until at the end of the game when one player picks the last one and wins.

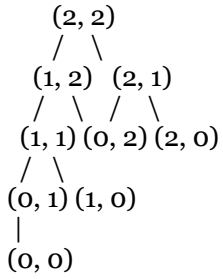
So, for player 1 to win, if we consider player 1 to start the game and he picks one stick from 1st pile. Now, if player 2 picks 2 sticks from pile 2 or 1 stick from pile 1, then player 1 wins in this case as one of the piles became empty.

Similarly for player 2 to win, if we consider player 1 to start the game and he picks 2 sticks from 1st pile. Now, player 2 wins in this case as one of the piles became empty. If player 1 picks 1 stick from pile 1, then for player 2 to win, he must pick one stick from pile 2, leaving player 1 to pick either one stick from pile 1 or 2, thereby letting player 2 win.

**(b)** [5 pts] Using your proposed presentation, draw the game tree for this game.

**Answer:**

The game tree can be represented in a way where each node would represent game state and the adjacent nodes(children's nodes) will represent the game states reachable from current state. The root represents the initial state (2,2). The end node depicts the final game state where one player has taken the last stick and won. An example for the same would look like :-

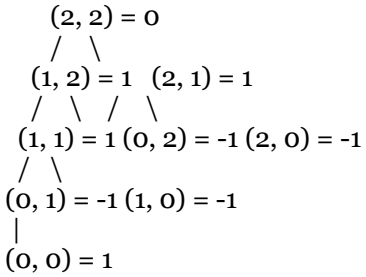



**(c)** [5 pts] Let a state that results in a win for the first player (MAX) be of value 1 while a state that results in a win the second player (MIN) be of value -1. Solve the game by assigning a value for each node in the game tree above. If you want to win, would you opt to move first or second? Do provide an explanation.

**Answer:**

We can solve the game tree problem, by assigning values to each node using minimax algorithm.

The game tree can be solved by assigning values to each node using a minimax algorithm. The method operates by giving the MAX nodes (nodes where the current player seeks to maximize the score) a value of 1 and the MIN nodes a value of -1. (Nodes where the current player aims to minimize the score). The algorithm climbs the tree starting at the leaf nodes (the final game states), giving each parent node a value based on the values of its offspring. A MAX node's value is equal to the highest value of its children, while a MIN node's value is equal to the lowest value of its children.
For the game tree above, the values of each node can be computed as follows:




Here, each node represents the best outcome for current player. The first player can assure a win by making the best move at each turn, therefore if a player wants to win, they will choose to go first. Only a draw will do for the second player. This is so that, although the second player can only reach a state with a value of -1 by making the optimal move, the first player can reach a state with a value of 1 by doing so. Therefore, if both players play their best, the first one to make a move will always win.

**(d)** [5 pts] Suppose instead of two piles of two sticks each, we have two piles of three sticks each. Now, would you opt to move first or second? Do provide an explanation.

**Answer:**

In case we have two piles of three sticks each, the ideal strategy will be to move second. Because second player can guarantee a win by playing optimally and the first one can only guarantee a draw by playing optimally.

Therefore, first player's best course of action will be to leave one stick in each pile , resulting in state (1,1). This is since, regardless of what the second player does, the first player can always react in a way that the game ends in a draw (1, 0) or (0, 1).

By selecting two sticks from one pile on the first turn of the second player, the state (1, 3) or (3, 1). To get to the end game state (0, 0), which results in a win for the second player, the second player can then proceed by making the best move during each turn.

As a result, the second player in this scenario would choose to move first to ensure a win while the first player would choose to move second to only ensure a tie.

## Q5. [15 pts] Heuristic-Guided Search

Consider the 8-puzzle that we discussed in class. We know that there are two admissible heuristic functions: (1) $h_1(n)$ denotes the number of misplaced tiles; and (2) $h_2(n)$ denotes the sum of distances of the tiles from their goal position. Suppose now we define two new heuristic functions: $h_3(n) = (h_1(n)+h_2(n))/2$ and $h_4(n) = h_1(n)+h_2(n)$.

**(a)** [10 pts] Is $h_3$ and $h_4$ admissible? Do provide an explanation.

**Answer:**

h3 and h4 are both admissible heuristics for the 8-puzzle problem.

Admissible heuristic never overestimates the actual cost of reaching the goal state. The value of the heuristic function is always less than or equal to the actual cost of reaching the goal state.

Because h3 is the average of two valid heuristics, h1 and h2, it is acceptable. H1 and H2 are both acceptable and never overstate the cost; h3, which is the average of the two, likewise never does.

Because h4 is the result of two valid heuristics, h1 and h2, it is acceptable. Since h1 and h2 are both acceptable and never exceed the real cost, their combined value, h4, will also never do so.

In conclusion, both h3 and h4 are admissible heuristics for the 8-puzzle problem.

**(b)** [5 pts] If admissible, compare their dominance with respect to $h_1$ and $h_2$.

**Answer:**

Without additional information, such as the actual costs of achieving the desired state and the values of the heuristics for a particular state, it is impossible to compare the dominance of h3 and h4 with respect to h1 and h2.

However, h4 is likely to be more valuable than h1 and h2 separately, and h3 is likely to be valued somewhere between h1 and h2. Because h3 is the average of h1 and h2 and h4 is the sum of h1 and h2, this is the case.

The effectiveness of an intelligent search algorithm can be impacted by the heuristic function selected. It's critical to thoroughly evaluate the traits of each heuristic and select the one that best fits the issue at hand.

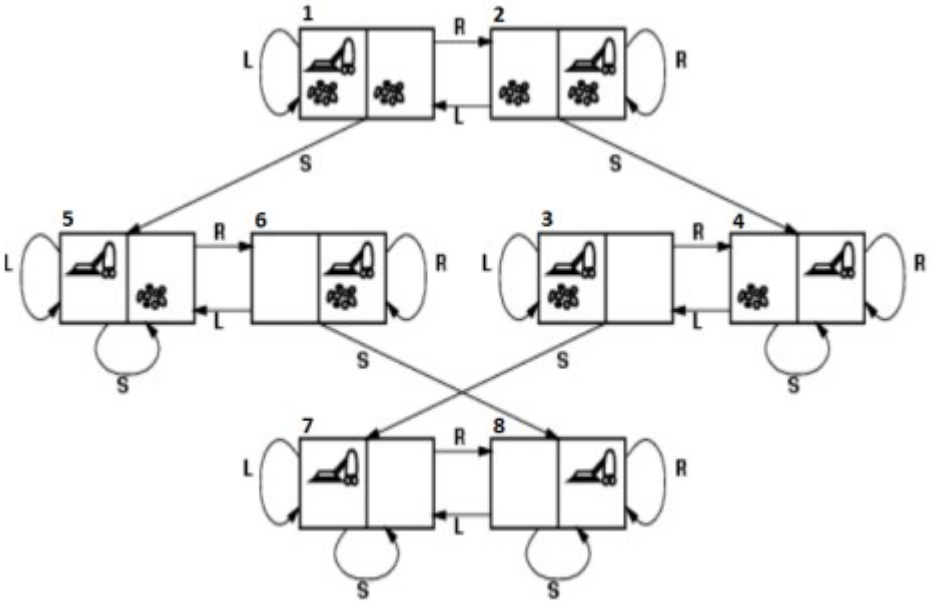



Figure 1: The state space of the vacuum world.