CSCI 4511W - Introduction to Artificial Intelligence
Homework 1
goelx029 | 5138568

Problem 1: For each of the scenarios below, classify the environment based on the seven classifications discussed in class (i.e. fully/partially observable, single/multi-agents, etc.). Additionally for each of the seven classifications, provide a single sentence supporting your reasoning.

1) **Sorry (Board Game)**
   a) **Fully v/s Partially Observable**
   Fully Observable Environment, the player can see the whole board at once and can see where the other person/agent's pieces are and what could their potential next move be depending on what potential cards they may draw.

   b) **Single v/s Multi Agent**
   Multi Agent Environment, there are two to four people playing Sorry at once. The movements of other players/agents greatly affect what action the agent should take when thinking about ways to win.

   c) **Deterministic v/s Stochastic**
   Stochastic Environment, because each move depends on which card a person draws from the deck of cards and there is a probability associated with each card draw hence each move has a probability of going to a particular place and hence the environment is stochastic and not deterministic.

   d) **Episodic v/s Sequential**
   Sequential Environment, because each action is dependent on the previous actions taken by the agent, as the new actions depend on the previous card draws and movement. This could be further said that because the number of cards for each type of possible move changes as cards are drawn from the deck which affects the possible sequence of moves in future (in a way the current move depends on previous move) hence making the environment sequential.

   According to books definition, the environment would be sequential because the current move by the player can change what would happen in the future, in terms of whether you move a particular pawn when you had a choice of moving one of multiple pawns.

   e) **Static v/s Dynamic**
   Static Environment, because nothing changes when the agent is thinking during its turn. Only a particular agent/person can perform an action at one time and all other players have to wait.

   f) **Discrete v/s Continuous**
   Discrete Environment, because the places the player's pawn can be is only at a certain places which is limited by the places marked on the board.

   g) **Known v/s Unknown**
   Known Environment, because it is assumed that the player knows the rules of the game and

knows what each action entails.

2) **Tennis**
   a) **Fully v/s Partially Observable**
   Fully Observable Environment, the agent can see the whole court at once (given they have enough power) and can see where the other person/agent is going or what the other person's/agent's next action would be.

   b) **Single v/s Multi Agent**
   Multi Agent Environment, there are two to four people playing tennis at once, most of the times competitively. The movements of other players/agents greatly affect what action the agent should take when thinking about ways to win.

   c) **Deterministic v/s Stochastic**
   Deterministic Environment, because the agent will always know what would happen when they take a certain action with respect to the ball. For Example, if the agent shoots the ball towards the right side of the court it will go towards the right side of the court.

   d) **Episodic v/s Sequential**
   Episodic Environment, because each action/shot is independent of the previous shot because it depends on where the ball is going after the opponent hit it. Also current move/shot does not have very profound effects on the game in the long term.

   e) **Static v/s Dynamic**
   Dynamic Environment, because the opponent and the ball is always moving when the agent is thinking.

   f) **Discrete v/s Continuous**
   Continuous Environment, because there is fluid and continuous change between the different states (movement of the ball and players).

   g) **Known v/s Unknown**
   Known Environment, because it is assumed that the agent knows the rules of the game and knows what will happen when the ball lands at a particular place and also what will happen when the ball is hit in a certain way.

3) **Exploring an undiscovered underwater cave system with a robot equipped with just a camera for sensing.**
   a) **Fully v/s Partially Observable**
   Partially Observable Environment, because the agent cannot see the whole cave at once or sense it at once, it can only see/sense a particular part of it hence partially observable

   b) **Single v/s Multi Agent**
   Single Agent Environment, the only agent in this environment is the robot which is using a camera for sensing the cave. There is no other agent interfering or part of the whole sensing action hence single agent.

c) **Deterministic v/s Stochastic**
Stochastic Environment, because there is always an associated probability with what could happen next when moving around the cave because it is undiscovered (no memory of what it is)

d) **Episodic v/s Sequential**
Sequential Environment, because the cave is like a maze and the movement of the robot depends on which movement was taken before. The path in the cave depends on where your current position is which is in turn dependent on your previous movements.

e) **Static v/s Dynamic**
Dynamic Environment, because of the water current and other possible creatures in the underwater cave.

f) **Discrete v/s Continuous**
Continuous Environment, because of the movement of the robot is continuous in the 3Dimenisonal Space of the underwater cave. Also as mentioned in the book, camera sensor is considered a continuous environment

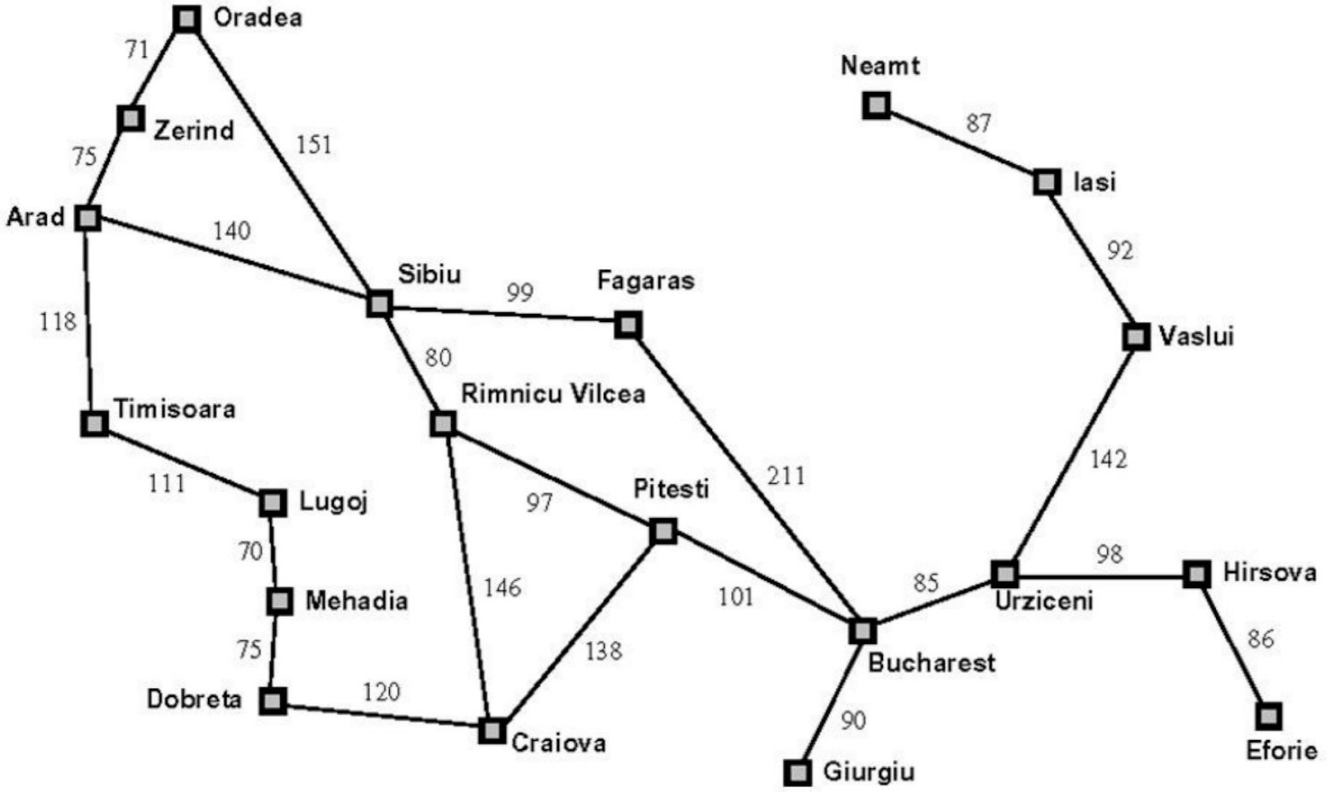
g) **Known v/s Unknown**
Unknown Environment, because the agent does not know what are the possible effects of its actions on the cave earlier, however they can be learnt from experience.

Problem 2: Consider the graph shown below. Assume you start at "Arad" and want to go to either "Dobreta" or "Urziceni". Show step-by-step how you would solve this with a breadth first graph search. When adding to the breadth first search queue, do it in lexicographical order (i.e. If you were starting at "Iasi" you would first add "Neamt" then "Vaslui" as N is before V in the alphabet). At each step show:

(1) the "fringe" nodes (the queue)

(2) the "explored" nodes (stuff that has left the queue)

(3) which node you are taking next from the fringe set to move to the explored set



I am assuming that the questions means that we can either go to Dobreta or Urziceni, meaning if we reach anyone of these places we are basically done and don't need to go any further

State 1: (Initial State)

Fringe - [Arad]

Explored - []

Moving from Fringe to Explored - Arad

State 2:

Fringe - [Sibiu, Timisoara, Zerind]

Explored - [Arad]

Moving from Fringe to Explored - Sibiu

State 3:

Fringe - [Timisoara, Zerind, Fagaras, Oradea, Rimnicu Vilcea]

Explored - [Arad, Sibiu]
Moving from Fringe to Explored - Timisoara

State 4:
Fringe - [Zerind, Fagaras, Oradea, Rimnicu Vilcea, Lugoj]
Explored - [Arad, Sibiu, Timisoara]
Moving from Fringe to Explored - Zerind

State 5:
Fringe - [Fagaras, Oradea, Rimnicu Vilcea, Lugoj]
Explored - [Arad, Sibiu, Timisoara, Zerind]
Moving from Fringe to Explored - Fagaras

State 6:
Fringe - [Oradea, Rimnicu Vilcea, Lugoj, Bucharest]
Explored - [Arad, Sibiu, Timisoara, Zerind, Fagaras]
Moving from Fringe to Explored - Oradea

State 7:
Fringe - [Rimnicu Vilcea, Lugoj, Bucharest]
Explored - [Arad, Sibiu, Timisoara, Zerind, Fagaras, Oradea]
Moving from Fringe to Explored - Rimnicu Vilcea

State 8:
Fringe - [Lugoj, Bucharest, Craiova, Pitesti]
Explored - [Arad, Sibiu, Timisoara, Zerind, Fagaras, Oradea, Rimnicu Vilcea]
Moving from Fringe to Explored - Lugoj

State 9:
Fringe - [Bucharest, Craiova, Pitesti, Mehadia]
Explored - [Arad, Sibiu, Timisoara, Zerind, Fagaras, Oradea, Rimnicu Vilcea, Lugoj]
Moving from Fringe to Explored - Bucharest

State 10:
Fringe - [Craiova, Pitesti, Mehadia, Giurgiu, Urziceni]
Explored - [Arad, Sibiu, Timisoara, Zerind, Fagaras, Oradea, Rimnicu Vilcea, Lugoj, Bucharest]
Moving from Fringe to Explored - Craiova

State 11:
Fringe - [Pitesti, Mehadia, Giurgiu, Urziceni, Dobreta]
Explored - [Arad, Sibiu, Timisoara, Zerind, Fagaras, Oradea, Rimnicu Vilcea, Lugoj, Bucharest, Craiova]
Moving from Fringe to Explored - Pitesti

State 12:
Fringe - [Mehadia, Giurgiu, Urziceni, Dobreta]
Explored - [Arad, Sibiu, Timisoara, Zerind, Fagaras, Oradea, Rimnicu Vilcea, Lugoj, Bucharest, Craiova, Pitesti]

Moving from Fringe to Explored - Mehadia

State 13:

Fringe - [Giurgiu, Urziceni, Dobreta]

Explored - [Arad, Sibiu, Timisoara, Zerind, Fagaras, Oradea, Rimnicu Vilcea, Lugoj, Bucharest, Craiova, Pitesti, Mehadia]

Moving from Fringe to Explored - Giurgiu

State 14:

Fringe - [Urziceni, Dobreta]

Explored - [Arad, Sibiu, Timisoara, Zerind, Fagaras, Oradea, Rimnicu Vilcea, Lugoj, Bucharest, Craiova, Pitesti, Mehadia, Giurgiu]

Moving from Fringe to Explored - Urziceni

State 15:

Fringe - [Dobreta]

Explored - [Arad, Sibiu, Timisoara, Zerind, Fagaras, Oradea, Rimnicu Vilcea, Lugoj, Bucharest, Craiova, Pitesti, Mehadia, Giurgiu, Urziceni]

Moving from Fringe to Explored - Dobreta

BFS Complete. Reached One of the two destinations - Urziceni

Problem 3:For each of the situations specify: (a) The initial state, (b) possible actions from the initial state, (c) a general description of other states, and (d) whether the approach is incremental or complete-state.

1) **You work at Amazon and someone placed a large order of stuff that you will have to pack into three separate boxes. Each box has a weight limit and a dimension limit that cannot be exceeded. (You do know that these three boxes are capable of storing all the items.)**
   The initial state is that all the boxes are empty and the items that have been ordered are still needed to be packed in the boxes.

   The possible actions for the next state would be to start putting the first item in one of the boxes and finding the boxes that could fit the first item. The possible actions could be categorized as the set of numbers - {1, 2, 3} where the number represents the box in which the current item will be put.

   The different states that could be reached includes first box being able to fit the first item or second box being able to fit the first item or third box being able to fit the first item and vice versa for each box. These states would then lead to states where we try to fit the second item in all of these boxes with first item fit in one of the boxes.

   The approach is an incremental approach because we are adding the items one by one and trying to determine whether they fit inside a box.

2) **You are imitating Google maps and want to find the shortest distance between two locations using roads.**
   The initial state is some 2 random places chosen on a map such that the current state is on the source location and the goal is to have the current state reach the destination location. In terms of a Graph, the source is marked as the start note and the node representing the destination is the goal that want to reach through the searching algorithm.

   The possible actions for the next state include adding all possible routes from the current state to the next possible state (small landmark nearby which leads to different routes). Kind of like can work on adding some distance in a particular direction - {N, NE, E, SE, S, SW, W, NW}

   The possible states could be different landmarks or intersections which lead to different ways or directions when trying to reach the destination. These states are a cause of change in the current action. Hence each new state represents a need for changing the action or changing the direction in which the person should move such that reaching the destination is easier.

   Thinking that the whole set of locations have been already incorporated in a graph, the problem becomes that we want to find the path (shortest) from source to destination, which makes this approach and problem solving complete state approach.

3) **You are trying to figure out how to win the game checkers**
   The initial state is the state representing how a checkers game is set up when a new game is started.

   The possible actions for any checker game is to go diagonally right one step, diagonally left one step, or go over opponents pieces. I am considering that when talking about the action of going over

opponents pieces include going over all possible (multiple) pieces. So the possible moves is given by the set - {LD, RD, O}

The possible state could be the different states of a checkers game. Each state represents the game board after someone has had their turn. Each state has a certain position for each piece and also the player who will be getting next move associated with it.

This is a complete-state approach because we are not adding pieces one by one and figuring out how to win, but we are using all the pieces at once and moving them to find a way to win in checkers. Solving this kind of a problem is like finding a solution state using a graph traversal algorithm.

Problem 4: Between depth first search, breadth first search and uniform-cost search, for each part say which search is most appropriate? Support your answer with one to two sentences explaining why this search is the best, along with a description of how you will represent the problem as a tree or graph.

1) **You want to open one of your locks (the spin-type shown below, where you need to enter 3 sets of numbers by spinning first clockwise, second counter-clockwise and third clockwise again) but you have forgotten the numbers and need to brute force it.**

I will use a Depth First Search to brute force the lock. I am using Depth First Search because it is the most practical way to actually find all the possible combinations and try them out. Because Depth First Search will traverse depth wise, during it's traversal we will always be at a state (considering a leaf node in DAG) such that the state represents the current three numbers that can be tried on the lock to see whether it works. Other algorithms like Breadth First Search and Uniform Cost Search does not actually lead to these states first, but will consider a number of incomplete numbers, for which we cannot really practically use them on the lock, as we know that it uses 3 digits.

To represent the possible states as a tree or a DAG, I will start with a null node (specifying no number) then add all the possibilities of the first number as the child nodes, then for each possible first number add all the possibilities of the second number as the children nodes to each first number node and similarly till we have three numbers represented as form of nodes (height of the tree - 4, including the root which is the null node)

2) **You go to the renaissance festival and walk through the hedge maze (https://en.wikipedia.org/wiki/Hedge_maze). You do not know the structure of the maze. How should you walk through the rooms to reach the end?**

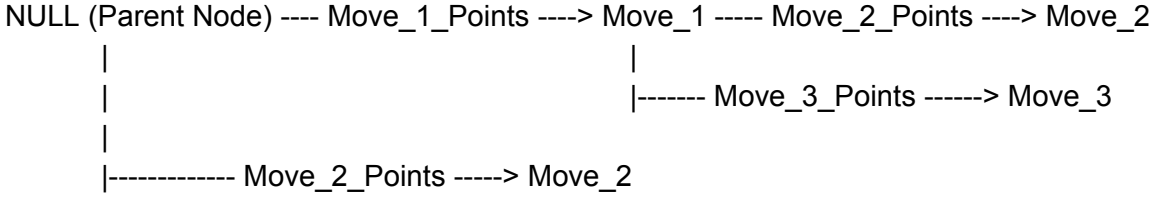
I will use a Depth First Search, to solve the Hedge Maze. I think Depth First Search is the best way because when trying to solve the maze it is important not to consider each and every move at every step but learn from mistakes and backpropagate back till the goal is reached. Because DFS is the kind of algorithm which will lead a person to actually start on a particular path and come back from mistakes until a solution is found it seems appropriate. Also considering that a person cannot actually compute all the possibilities in a maze because there is only one of him, DFS seems to be the better choice in terms of thinking of implementing it as well.

To represent a maze as a tree or a graph, we can consider each turn, intersection and dead end in the

maze as a node/vertex and the paths between these different turn, intersection and dead end that connect them as the corresponding edges that connect the vertices. Then the structure we will get becomes a graph representation of the maze and applying a DFS algorithm will give us the ways to take to actually reach the end.

3) **You are performing a dance routine and know how to execute 20 different moves. Each move uses a certain set of muscles and receives a fixed amount of points. You have time to execute 5 moves during your performance, and you want to get the most points possible (without overusing the same muscles too much).**

Considering that each move has a fixed set of muscles and it does not depend on which move was executed before (as in that the muscles which were used does not depend on from which move to which move did the dancer perform). I will use a uniform cost search method with a change in underlying algorithm to use a maximum (max heap) based priority queue such that the path that the algorithm finds is the longest path, and the edge weights that I will use in the graph will correspond to the number of points for doing a particular move. To further elaborate, I will first make a graph such that it starts from a null node, then from the null node, I add the possible dance moves as children nodes such that the children nodes will only be added if the frequency of muscle used after performing the child move for each muscle is less than some threshold. This will guarantee that the possible paths with the overuse of muscles are not considered. Following this strategy a tree of max height (6, considering the null node) will be constructed. Then each edge weight will be the number of points for the next (child) move. Hence a small part of the graph possible look at the top layer would be something like:



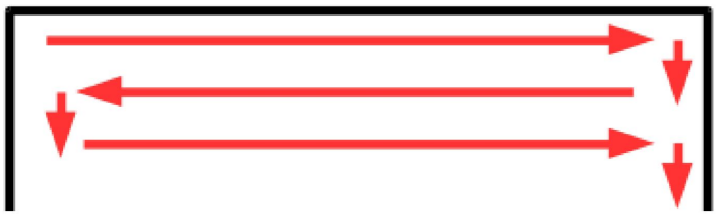
Problem 5: For each of the following, state whether the task is being done rationally. (Note: this is the strict artificial intelligence definition of "rational".)

1) **A human playing the card game "War" (someone else deals the cards) (Game rules: https://www.pagat.com/war/war.html)**
   Considering that the goal of the person is to win the game and that the person cannot really do something except pick up a card from the face down deck and hope for the best, I think the person is playing the game rationally as each action is the only possible best action because the person is restricted by what they can do each chance.

   Originally I thought that the human was not rational but I changed my understanding because I think that according to book, Rationality means: "**For each possible percept sequence**……". Hence when putting into perspective that the human player is limited by what they can actually do each move, it is the best outcome every time that could be hoped for.

2) **A robot looks at historical trends on the stock market to try an invest to get you the most money.**
   Considering that the performance measure is getting the most money, it seems that the robot is rational, because it is using knowledge extracted from historical data and is applying that knowledge to actually get the best outcome (possible). It is still probable that the robot might not get the most money, but the action it took certainly based on the history was expected to which makes it rational.

3) **You want to find a treasure buried square area as fast as possible. All you have to detect the treasure is a metal detector, so you "zig-zag" (go perpendicular to an edge until you reach the other side, the move the furthest away to ensure you do not miss anything with the metal detector and go back (again perpendicular)) as shown below until the metal detector finds the treasure. Once you have located the treasure you dig it out.**



   I am considering that the treasure is present in the square area which is same as the area covered by the metal detector and that the metal detector will only beep when it is actually on top of the square area with treasure.
   Considering that the goal of the agent is to find the treasure as early as possible and that the agent does not have past data, the current form of action that the agent takes is the rational decision because it is doing an exhaustive search of all the single square areas in which treasure could be present. And the way the agent is working will guarantee finding the treasure in expected minimum time possible.

   However if the area occupied by the Treasure is much bigger than the area of the metal detector and still less than the overall area of the land, this agent is not rational, because it should change the trajectory based on when the metal detector does not beep after beeping once. This will help find the

right area (collection of squares).

Also if the metal detector works based on proximity, the agent is not rational, because the direction in which the agent should go to depends on where the first detection happened on metal detector and based on the intensity/confidence make sure to find the correct direction to go to.