Patrick Mejia
Alex Alvarez
Madeline Chudy
Dylan Phillips

Group Assignment 1

Question 1:

1. An agent that senses only partial information about the state cannot be perfectly rational.
False because perfect rationality is achieved when the agent selects the best actions that can maximize the expected usage, as long as it makes rational decisions. if there are unexpected obstacles that the agent was not trained for then such as a self driving car having a tree branch fall in the road unexpectedly, the car is still perfectly rational because it detected what it was supposed to detect but there were outside factors

2. There exist task environments in which no pure reflex agent can behave rationally.
True because there are some tasks that involve thinking multiple steps ahead but the pure reflex agent would only react to the current state of the task. An example of this would be a chess game because the reflex agent would only react to the current state of the board, while a chess master already has 3-4 steps ahead

3. There exists a task environment in which every agent is rational.
False because there can be some task environments that are very unpredictable such as a dice roll. A dice roll game relies on probability and there is no way for an agent to be able to get the correct prediction for the game besides on probability

4.The input to an agent program is the same as the input to the agent function.
False this is because the input to the agent function is usually a single percept or sequence that the agent receives and then it produces an output. The input to the

agent program would be the percepts but could also be other information that the program needs in order to function

5. Every agent function is implementable by some program/machine combination.
True This is because its always possible to write a program in some programming language and execute it on a computing device that correctly implements the agent function

6. Suppose an agent selects its action uniformly at random from the set of possible actions. There exists a deterministic task environment in which this agent is rational.
False if the agent selects actions at random, it would not be considered rational because an agent selects the tasks based on probability to maximize its usage based on the data it was trained on. If the agent decides to randomly select the task, there is a chance it selects the optimal solution, but its likely it would not be the correct one, which means it would not be rational

7. It is possible for a given agent to be perfectly rational in two distinct task environments.
True its possible that it can be perfectly rational in terms making optimal decisions if it is given the clear objectives, it can use its data to make optimal decisions in two distinct task environments. If the agent is designed to be able to reason and make optimal decisions in different types of environments, then it could be perfectly rational in two distinct task environments

8. Every agent is rational in an unobservable environment.
False , this is because the agent must be able to use the information and its sensors to learn about the environment it is in to understand the state of it. Without the ability to do so, the agent would not be able to make optimal decisions. The only way to solve this issue would be to design an agent that can make decisions on the small amount of information given to it and continue to learn about the environment based on the feedback it is given.

9. A perfectly rational poker playing agent never loses.

False. The perfectly rational poker playing agent would win more than most opponents but poker is also based on probability and has a lot of incomplete information which means it would be very unlikely for a poker playing agent to never lose strictly on probability

Question 2:

Your goal is to navigate a robot out of a maze. The robot starts in the center of the maze facing north. You can turn the robot to face north, east, south, or west. You can direct the robot to move forward a certain distance, although it will stop before hitting a wall.

1. Formulate this problem based upon the above observation.

State Space: The state space is defined as a coordinate system with the center of the maze at (0,0). The maze itself is a square spanning from (-1,-1) to (1,1). There are n possible locations, 4 possible directions to move in – therefore 4n possibly states.

Initial State: The initial state is the robot at coordinate (0,0) facing North.

Goal Test: For this problem, the goal test is to exit the maze, and either $|x| > 1$ or $|y| > 1$ where (x, y) is the current location.

Possible Actions: The possible actions are for the robot to face North, face South, face East, face West, move, stop.

Successor Function (Transition Model): Move towards any distance d and change the direction in which the robot faces. (In (state), face(x)), where x is North, South, East or West.

Cost Function: The cost is the total distance moved.

2. In navigating a maze, the only place we need to turn is at the intersection of two or more corridors. Reformulate this problem using this additional observation.

The intersection the robot is currently at as well as the direction it is facing will be recorded by the state. An exit node exists at the end of each corridor leaving the maze. Assume that a node corresponds to the center of the maze.

State Space: In total there are now 4i + 2(n – i) states, where i is the number of intersections and n is the total number of states. Let the locations of intersections of corridors be i, different from any location in the state (represented by n in the previous part). The state space now is 4i instead of 4n. Since i is within n (intersections are part of the total possible states) the space where the robot will NOT stop is represented by (n – i). There are two options in this space, move forward or stop when reaching a wall. Therefore, this is represented in the state space by 2(n – i). The total state space is now 4i + 2(n – i).

Initial State: The initial state is the robot in the center of the maze facing North.

Goal Test: The goal test is for the robot to be at an exit node.

Possible Actions: The possible actions are for the robot to face North, face South, face East, face West, move, stop.

Successor Function (Transition Model): The robot moves to next adjacent intersection (in front of it – if there is one), turns to face new direction. (In (state), face(x)), where x is North, South, East or West.

Cost Function: The cost is the total distance moved.

3. From each point in the maze, we can move in any of the four directions until we reach a turning point, and this is the only action we need to do. Reformulate the problem using these additional actions.

State Space: Since we no longer need to keep track of the robot's orientation, the 4 and 2(n – i) can be removed from the previous state space. This is because the 4 represented the possible directions the robot could face, and the 2(n – i) represented the non-intersection locations which can be excluded from the problem now. Therefore, the state space is i.

Initial State: The initial state is the robot in the center of the maze facing North.

Goal Test: The goal test is for the robot to be at an exit node.

Possible Actions: Move, stop. Once stopped – turn North, South, East, or West.

Successor Function (Transition Model): The robot moves until it reaches a wall. Move to the next intersection (if reached the North wall, get to the East, West, or South wall).

Cost Function: The cost is the total distance moved.