**Project Two**

Justin Starr

Department of STEM

CS 370 – Current/Emerging Trends in CS

Professor Jason Richard

June 23, 2024

**Project Two**

As an AI developer for a gaming company, it has been asked to design an intelligent agent for an NPC (non-player character) to represent a pirate in a treasure hunt game (maze) where the player needs to find the treasure before the pirate finds it. The pirate needs to navigate the game world, which consists of different pathways and obstacles to find the treasure, and the pirate agent’s goal is to find the treasure before the human player. The following is a design defense demonstrating the understanding of the fundamental AI concepts involved in creating and training the intelligent agent. The differences between human and machine approaches to solving problems will be analyzed, the purpose of the intelligent agent in pathfinding will be assessed, and the use of algorithms to solve complex problems will be evaluated.

We begin by describing the steps a human being would take to solve this maze. At the start of the maze, a human would begin by traversing various pathways with the goal of reaching the end of the maze. While traversing pathways a human may encounter obstacles and find it necessary to backtrack to the location of the maze where the human chose the pathway that led them to the obstacle. The human would then choose a different pathway and continue the process until the end of the maze has been reached. Along the way, the human would likely use some sort of strategy, such as making only right turns throughout the maze to help them navigate the maze more easily and also help them remember where they have and have not been in the maze.

The steps the intelligent agent takes to solve this pathfinding problem are similar, however, it requires much more learning of the environment and what it should and should not do to accomplish its goal. The agent begins at the beginning of the maze, and at first, it has no knowledge of the environment. At each game state, the agent chooses an action and then performs it. Then, based on the result of that action, the agent receives a reward, which can be a positive, negative, or no reward depending on how good or bad the action is that is taken by the agent. Future rewards are calculated at each game state to determine what reward the agent will get if it takes that action at that state in the future (Lamba, 2018). This is an iterative process (the game is played multiple times) which helps encourage the agent to reach the highest cumulative reward possible and reach the end of the maze faster. At first, a higher level of exploration is involved (more random actions are taken), however, the more the agent learns about game states and rewards, the less it explores, and it starts exploiting the environment, taking fewer random actions (Lamba, 2018). Also, at each state of the game, the agent remembers the experience and performs an experience replay where the agent samples experiences from memory (Surma, 2018).

The similarities and differences between the steps a human and the agent take to solve the maze are that at first, neither has any knowledge of the environment, however, the human may have some knowledge of the structure of the maze. Both the human and the agent take actions in the game environment, and similarly receive rewards based on their actions. While the agent’s rewards are based on a point system, the rewards for humans are similar in that a human might feel more rewarded or accomplished the closer the human gets to reaching the end of the game. Also, if a human would run into an obstacle, they might feel less rewarded because they would have to turn around. The idea in reinforcement learning where the agent is rewarded is intended to be very much lifelike (Surma, 2018). A significant difference is that humans can play the game more intuitively and it is likely that that humans would learn to avoid obstacles faster than the agent would until the agent is trained better. This required the agent to play the game multiple times and learn from its experience replay.

To assess the purpose of the intelligent agent in pathfinding, it is essential to understand the difference between exploitation and exploration, and what the ideal proportion is for this pathfinding problem. In reinforcement learning, exploration means learning about the environment the agent is acting in, and actions that are selected have uncertain outcomes in order to learn about the possible rewards that the agent will receive for taking those actions at those particular states of the game (GeeksforGeeks, 2024). Exploitation is where the agent begins taking actions based on the knowledge it has already acquired to maximize the expected reward (GeeksforGeeks, 2024). Essentially, along with maximizing rewards, this enabled the agent to become more efficient in its decision-making process, and it helps the agent avoid more obstacles by relying on information about actions it has already taken instead of relying on the uncertainty of actions that may be more unfamiliar to the agent (GeeksforGeeks, 2024). Determining the ideal proportion between exploration and exploitation is an essential aspect that must be kept in mind. While it is essential that the agent is able to learn by exploring, if the agent is never exploiting the environment, the agent will never try to get better at completing the game. Therefore, there must exist a balance between when the agent has explored enough that it can start taking actions based on what it already knows instead of uncertain actions. In the game that has been developed, the agent starts exploiting itself more and more as the win rate for the agent increases, which is an indication that the agent has learned enough from its previous actions and can start making better predictions on its own.

Reinforcement learning can help to determine the path to the goal (the treasure) by the agent (the pirate) by essentially applying a deep Q-Learning algorithm to the agent’s model where the agent learns episodically about the environment, its actions, and rewards at each game state, which help it determine the actions it should take in the future. Eventually, after learning what rewards the agent will receive at particular states of the game, the agent can then learn to reach its goal or reach the end of the maze by seeking the highest future rewards possible.

The way deep Q-learning was implemented using neural networks for this game was by first building the neural network with three dense layers and using PreLU as the activation, adam as the optimizer, and using the mean square error as the loss function. When combined these are used to train the model. Then, a deep Q-learning algorithm is implemented to find the best possible navigation sequence that results in reaching the treasure cell while maximizing the reward. It is implemented so that the agent can choose the best possible action for a given circumstance (observation), where each observation has its own Q value, the quality of any given move (Surma, 2018). The deep Q-learning algorithm iterates for the number of epochs defined, and for every epoch, a new game is played. In each game, the agent randomly selects a free or empty cell and the maze is reset with the agent placed at that position. The environment state is the current state of the maze configuration where the agent currently is. A game then begins to track each episode (game state) until the game is over. While the game is not over, the agent selects a valid action either by exploration or exploitation depending on how much the agent has already learned. It then takes the selected action where the environment state, reward, and game status are determined based on the action taken. If the returned game status is over, then the game loop ends, if not it repeats the process of taking an action and determining the game state, reward, and game status. Each of these iterations is an episode that gets stored into an experience replay object where previous actions are remembered to help the agent predict better future actions based on the highest future cumulative rewards. The neural network is then trained and loss is determined. Once the game is over, if the win rate is above the specified threshold, and the model passes the completion check, then that single epoch is over. The completion check is a check to determine if training has exhausted all free cells and if in all cases the agent won. By implementing deep Q-learning for this problem, we were able to have the agent successfully learn how to solve a complex problem by enabling it to learn from previous actions and maximizing future rewards, which ultimately led to the agent successfully learning how to play the game on its own.

**References**

GeeksforGeeks. (2024, May 18). *Exploitation and Exploration in Machine*

*Learning*. <https://www.geeksforgeeks.org/exploitation-and-exploration-in-machine-learning/#>

Lamba, A. (2018, September 3). *An introduction to Q-Learning: reinforcement*

*learning*. Medium. <https://medium.com/free-code-camp/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc>

Surma, G. (2018, September 26). *Cartpole - Introduction to Reinforcement Learning (DQN –*

*Deep Q-Learning)*. Medium. <https://gsurma.medium.com/cartpole-introduction-to-reinforcement-learning-ed0eb5b58288>