**Project Two**

Thi Hong Van Nguyen

Department of Computer Science and Engineering, Southern New Hampshire University

CS-370 - Current/Emerging Trends in CS

Timothy Alexander

October 15, 2023

**Differences between human and machine approaches to solving problems**

In this treasure hunt game, the approach to solving problems differs between humans and machines. Typically, when humans solve such a maze, they visually inspect the maze to understand its structure, identify the goal (treasure) and obstacles, and mentally plan a path while considering obstacles and possible actions. They navigate the maze, making decisions at intersections based on their current knowledge. If they encounter obstacles or dead ends, they backtrack and explore alternative routes. Humans continue navigating and updating their mental map of the maze until they reach the treasure or determine it's impossible.

On the other hand, the intelligent agent is using a method called Q-learning to figure out the best way to find the treasure in the maze. It starts by setting up a table to keep track of how good different actions are in different situations. Next, it explores the maze by either trying things randomly or by picking the action that seems the best. As it moves through the maze, it remembers what it is doing, what it gets as a reward, and what happens next. Then, it uses this information to make its table better. It learns from its experiences to get better at picking the right actions. This process happens many times, and over time, the agent gets really good at finding the treasure in the maze.

While both humans and the intelligent agent aim to find a path to the treasure, their approaches differ significantly. Humans rely on innate reasoning and problem-solving capabilities and can adapt to various scenarios. The intelligent agent, on the other hand, is specialized for solving the specific treasure hunt problem and relies on a trained model and predefined algorithms. Additionally, humans use visual perception and mental mapping, while the agent relies on discrete states and actions in a grid-like maze.

**Purpose of the intelligent agent in pathfinding**

The roles of exploitation and exploration are critical for effectively solving complex navigation problems like finding the shortest path or reaching a goal in this Treasure Hunt Game. Exploration involves trying out different actions or paths to gain more information about the environment, while exploitation involves choosing actions believed to be the best based on past knowledge (Coggan, 2004). In this pathfinding problem, the balance is determined by the exploration factor, epsilon, set to 0.1 in the code. This means that the agent randomly explores about 10% of the time and exploits the best-known actions for the remaining 90%, enabling the agent to learn new paths while primarily relying on existing knowledge and maintaining a balance between experience and informed decisions. This proportion of exploitation and exploration allows the agent to learn and adapt to different maze configurations and discover the most efficient path to the treasure. By exploring a fraction of the time, the agent can gather new experiences and avoid getting stuck in suboptimal paths. However, by primarily exploiting its current knowledge, it leverages its learned information to make informed decisions and steadily improve its pathfinding abilities over time.

Besides the initial choice of 0.1 as the epsilon value, I conducted two experiments with epsilon values of 0.2 and 0.05, respectively. The results are as follows:

**Epsilon 0.1**

Reached 100% win rate at epoch: 156

n\_epoch: 156, max\_mem: 512, data: 32, time: 32.65 minutes

**Epsilon 0.2**

Reached 100% win rate at epoch: 74

n\_epoch: 74, max\_mem: 512, data: 32, time: 20.91 minutes

**Epsilon 0.05**

Reached 100% win rate at epoch: 81

n\_epoch: 81, max\_mem: 512, data: 32, time: 20.14 minutes

To determine which epsilon value gives the best result, I consider different factors: win rate, training time, number of epochs and resource usage. Regarding win rate, all three epsilon values resulted in a 100% win rate, so they are equally effective in terms of solving the maze. However, we should consider whether the win rate stabilizes at 100% in later epochs or if there is any variance. From training time perspective, epsilon 0.1 had the longest training time, followed by epsilon 0.2 and then epsilon 0.05. If a more time-efficient solution is desired, epsilon 0.2 or 0.05 might be preferred. Similarly, epsilon 0.1 required the most epochs to reach a 100% win rate, while epsilon 0.2 and 0.05 reached it faster, so if we want the model to converge more quickly, epsilon 0.2 or 0.05 might be better. Lastly, in terms of resource usage, epsilon 0.1 might consume more resources due to the longer training time and more epochs. Ultimately, the choice of epsilon depends on our priorities. In this case, I would choose epsilon 0.2 as it offers the shortest training time and fewer epochs.

Reinforcement learning, in this game context, enables the pirate agent to determine the optimal path to the treasure in the maze. It balances exploration and exploitation through an epsilon-greedy strategy, preventing the agent from getting stuck in suboptimal paths while exploiting the best-known routes as mentioned above. The agent collects experiences, including observations, actions, rewards, and states, and employs a deep Q-learning model to approximate the expected rewards for actions in different states. Through iterative training, the model refines its Q-value estimates, enabling efficient decision-making. Ultimately, the agent aims to win the game by leveraging the learned policy to navigate the maze and successfully reach the treasure cell.

**The use of algorithms to solve complex problems**

Algorithms were utilized to tackle intricate problems. In this context, Q-learning algorithm was employed through the use of neural networks to address the complex task of guiding a pirate to find treasure within a maze. The agent is represented by a neural network model. The architecture of the model consists of multiple layers, including dense layers. These layers serve as the Q-table, containing estimates of the expected rewards for various actions in different states of the maze. The neural network is trained to improve its Q-values based on its interactions with the environment. The agent explores the maze, choosing actions based on either random exploration or by exploiting its current knowledge (Q-values). During exploration, it gathers experiences, including its current state, chosen actions, rewards, and the resulting state. These experiences are then used to update the Q-values through a supervised learning process. This iterative training continues over multiple epochs, enhancing the agent's decision-making abilities and enabling it to determine the most effective path to reach the treasure in the maze. Through this implementation, the agent learns and adapts its actions based on feedback received in the form of rewards and punishments, ultimately enabling it to navigate complex mazes and locate the treasure.

**References**

Coggan, M. (2004). *Exploration and Exploitation in Reinforcement Learning*. Computing Research Association. chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/<http://archive2.cra.org/Activities/craw_archive/dmp/awards/2004/Coggan/FinalReport.pdf>