RL Project Proposal - Frozen Lake

Jaiden Callies Alex Hunt

For our reinforcement learning project, we have chosen to create an agent to play Frozen Lake, a game aimed at obstacle avoidance and maze traversal. We hypothesize that we can create and train an agent to successfully complete the Frozen Lake maze.

Frozen Lake is a single-player arcade game. The goal of the game is to maneuver an elf character across a frozen lake to reach its present. The game environment is essentially a 4x4 two-dimensional grid with 16 spaces. There is exactly one starting space and one goal space. All other spaces represent either frozen portions of the lake or holes within the lake. To win the game, the player must reach the goal state by traversing the lake without falling into one of the holes. If the player falls into one of the holes, the player loses. A player may move one space at a time either left, right, up, or down across the grid. In order to increase the game's difficulty, there is an optional feature to make the lake slippery. If the lake is slippery, the actual movements of the game's character may not mimic the player's chosen actions. We will likely experiment both with and without this feature.

We will use different reinforcement learning methods to train our agent. Jaiden will implement dynamic programming and Monte Carlo learning methods. Alex will implement temporal difference learning and eligibility traces. There will be some overlap between our approaches. For example, for the sake of efficiency, we both intend on utilizing bootstrapping to train our agents. Jaiden will implement this technique through dynamic programming methods, and Alex will implement it using temporal difference learning.

The manner in which our agent will be rewarded during the learning process is also significant. With each time tick acting as a negative reward and the ice holes acting as a negative infinity reward, the agent will recursively choose the greatest integer value path solution. This is possible since the goal state would have a large positive value and the algorithm would choose the path with the greatest overall reward, which would end up being the optimal solution if a solution is available. However, the slippery feature in the game complicates this issue. Any given action may not always result in the intended outcome. Therefore, it is quite possible for an agent to execute an otherwise correct move toward the goal state and still end up falling into a hole.

There are a few different ways we could potentially evaluate our agent. The goal is to maximize the cumulative reward. Therefore, the most critical assessment will likely be to compare the optimal solution returned by the algorithm with the maze's actual optimal solution as determined by us. Another potential way to assess our agent could be to record the average number of steps taken when a game is won or lost. A player is not required to complete the maze in a given number of steps, so the number of steps is irrelevant to the general game format. However, it could nevertheless be used as a measure of the agent's efficiency.

There are a variety of ways to determine the efficacy of our project. For example, the average accumulated reward could be paramount in monitoring overall improvement. If improvements over time begin to dwindle, this may indicate that the current policy is optimal. Additionally, different learning methods have different convergence properties. Depending on the specific learning method used, it may be helpful to check whether certain values, such as value functions, stay stagnant. A minimal change in values may indicate convergence to an optimal policy.

Lastly, we plan to use the Frozen Lake interface from Gymnasium to train our agent. More information is available at `https://www.gymlibrary.dev/environments/toy_text/frozen_lake/`.