Agent 1: Rule Based Agent

# Early Versions

In its early versions, our rule-based agent was a very naïve agent. Prior to implementing the observation making code that was provided, we didn’t really have much knowledge on making observations of the gym environment. What we did found out through research (find links and insert them here) however, was the different types of moves that Mario could make. For example, calling:

Env.step(0) makes Mario do nothing.

Env.step(1) makes Mario move right etc.

Using this knowledge, we devised a very simple agent that would run a move right and jump action in a loop, making Mario continuously jump throughout the first level.

[INSERT CODE HERE]

We implemented this using a counter variable where while the counter variable was less than or equal to 16, Mario would continue to perform action 4 (run and jump right). When the counter reaches 17, Mario does action 0 (stop), and then starts the run and jump action again for another 16 steps. In other words, Mario performs action 4 for 16 frames and action 0 for 1 frame in a continuous loop.

The reason why the agent alternates between the actions is because changing from action 4 to an action that does not involve jumping represents letting go of the jump button, which enables Mario to jump again.

The value of 16 for the counter variable ceiling was reached simply through trial and error of various values. By trying numbers within the range of 1-20 we found that 16 allowed Mario to perform a full jump.

# Implementing Observations

The early version for this agent worked for the first level, but no other. So, we had to then use the observation making code to enhance its performance. Our work on the naïve algorithm didn’t go to waste as it allowed us to find out how to make the agent perform a full jump.

After implementing the ability to make observations, we distinguished between three types of jumps the agent could perform: an enemy jump, a block jump (over a stack of blocks or pipe etc.,), and a pit jump (over an empty space Mario could fall into).

Enemy Jump – The agent performs this by seeing if there is an enemy ahead (within reasonable distance), and then jumps over it.

Block Jump – The agent performs this by observing that there is a set of blocks or a pipe in front of Mario.

Pit Jump – This one is more complicated. The agent will first check if Mario is standing on a block. Then, if he is standing on a block, the agent will check if there is an empty block 1-2 blocks away from the one Mario is standing on. If that is the case, Mario will jump over the pit.

Performing these three types of jumps also involves once again a counter variable, same approach as the early version. However, to avoid Mario beginning to perform one jump, and then another type of jump overriding the initial one, we keep track of whether Mario is currently performing a jump or not (through previous\_jump). If previous\_jump is negative, it means the agent is not currently performing a jump and can make observations about its environment to determine whether it needs to perform a type of jump. If previous\_jump is not negative, the agent will perform the type of jump stored in the variable.