James Alford-Golojuch

CS463G

November 4, 2016

Prolog Assignment 2

Code Description:

For this second assignment I took the general idea of going from a start goal towards an ending goal from the first assignment. Instead of hard coding it to go towards a specific coordinate though I decided to have the path continue to expand until a certain item was found or goal achieved, or until every space that could be visited had been visited and the goal or item was not able to be achieved or found.

My program first looks to make sure at least one of every item is located in the maze so that it doesn't have to waste time trying to find items that aren't there. This isn't necessary but I added it since it may speed up the program a little. First the program attempts to find a path to an egg and plus it needs to hatch that egg by moving at least three spaces away from the spot the egg was picked up at. Next my program attempts to find a path starting from the last coordinates of the path it left off to the next item which in this case is a Pikachu. It is important to note that the order of the first 3 paths of items does not matter and if desired could easily switch around the order of which item path is found first. It repeats this starting where the last path left off for each item until we reach the mewtwo space with at least one of every item though it is possible to walk over and pick up more than one of each item. If a path is not found for one of the items then it returns false since no path exists to find all items and then capture mewtwo. I also created predicates which handle the checking of how many eggs have been hatched (hatchedeggs) and ones for incrementing the distance traveled for each egg that has been picked up until that egg has hatched (hatchingeggs).

Learning Outcome:

I learned that breaking apart a problem is so much easier than attacking the full problem at once. Initially I attempted to have the program continue searching the entire maze until a path is formed that has one of each item and ends on the mewtwo space. This proved to be a problem for me since I needed to set a max times that one space could be visited such that the program path doesn't get stuck in a loop. Even with a low amount of max visits the program seemed to take a significant amount of time. This made testing impossible since I could rarely find a solution especially for cases where an item was in the maze but unreachable. This lead me to my current approach of finding one path without revisiting a spot from the current point to the item. This got rid of my need for setting max space visits in the map while still allowing the program to visit previous spaces if needed to reach another different item.