Comp341 Hw2
Can Gözpınar 68965
Report

Q1) I tried using cumulative manhattan distance to all foods but it didn't improve much and resulted in some loops so I decided not utilizing it after all. I used successorGameState.win() to check to see if eating the food would result in an ultimate win state. I returned a very high value for that so that even if eating the last food gets me nearer to a ghost it still should be eaten. I used action to see if it is same as stop action and penalized stop action with a high negative return value so that, pacman doesn't get stuck and lie still in positions. I used manhattan distance to the closest food to the successor state by taking its negative. This lead pacman to choose states which it chases closest foods and eats them. I used the difference between the total number of foods in the current state and the successor state and returned it a positive high value to favor eating foods. I used negative of cumulative manhattan distance to ghosts from pacman. I returned lower values whenever ghosts got closer and if ghosts had no walls between the pacman I even lowered it more to make pacman hide from ghosts from dire situations. I think using negative and reciprocals is a good idea even though its far from perfect I guess. If values are well thought I believe they can be quite effective to navigate between situations such as our pacman world.

Q2) In my tests I observed a significant speed improvement in between the algorithms minimax and alpha-beta pruning algorithms. Alpha-beta pruning algorithm performed faster than minimax algorithm during my tests. It did faster in situations as specially when it was able to prune the search tree which gave it its advantage of time complexity over the minimax algorithm. Especially when the there were moves which would resulted in obvious advantage of the pacman like when it moved further from ghosts and had food closer to it at the same side whereas other side of the pacman did not fulfill these criterion in any way.

Q3) It did not behave exactly the same way because, pruning might lead to not discovering the best state. In alpha-beta pruning algorithm whenever the criterion for alpha is greater than beta is satisfied we do pruning. This leads our algorithm to miss on possibly better states which doesn't promise optimality whereas, minimax algorithm does promise optimality regarding the search depth limit that it works in our case.

Q4) With minimax and alpha-beta pruning algorithms there are occasions where our pacman agent runs towards the ghosts to die faster since living has a penalty for those agents. This behavior happens since those two algorithms assume that the ghost agents are optimal so whenever, there is no inevitable escape from loosing our pacman agent chooses the actions which leads to a quicker death. Also, whenever the ghost's has a chance to kill when moved optimally our pacman agent doesn't take risks. Forexample when there is food present between ghosts our pacman agent won't try to sneak in between the ghosts and eat those food and leave because those algorithms assume optimal ghost agents. On the other hand, expectimax algorithm takes risks from time to time since, it doesn't assume the ghost agents to be optimal but moves according to the best choice probability wise. This leads pacman to sneak in and try to eat food in between ghosts in cases which, the other two algorithms would not choose to do. Expectimax might lead to a quicker win sometimes but might loose more too.

Q5) On the better evaluation function, my evaluations had to be state based whereas before, we did it on action basis. I had put a lot of time when I started doing project to each question to master it to my best ability so therefore by utilizing the ideas I had already implemented in the first question I did well on this question too. I had to change the implementation of things and logic of some funcitons such as not comparing the number of foods on the pacman grid but just taking the number of foods on the grid alone into account. I had to convert some of my functions to work with taking only the current state instead of taking an action as it did for the solution of the first question.

Q6) I prioritized states that got me win, in other words if it got me closer to ghosts or danger did not matter if the state would have resulted in a win whereas, if it were not to be a win state then those things I’ve mentioned would have been taken into account. Number of foods on the pacman grid was something pacman had to minimize to win the game so it was an indication of the betterness of the state. Closeness to food were another thing which lead pacman to favor the states that closer to food. Surviving and not dying is a big part of the pacman game therefore increasing the manhattan distance to ghosts and penalizing closeness to ghosts whenever there were a single path without including a wall was a sign of great danger which had to be taken into account and penalize the return value of the state. So overall, I tried to favor the good behavior such as eating and chasing foods by increasing the return value of the state accordingly and, I tried to avoid situations that was unwanted such as not taking the shortest paths between eating foods and dying due to ghosts by, decrementing the return value of the state to lower the chance of that state being chosen over a more favorable state. For tuning the weight, I had to make sure that winning state and loosing state overruled all the other states on importance. Also I had to make sure that risk of dying by ghosts had higher weights than closeness to food did so I choose my weights accordingly to avoid risky situations by having higher weights for ghost and taking risks whenever the situation was not dire accordingly.