Orçun Özdemir

54020

COMP 341 Homework 2 Report

1. The reflect agent that I wrote will move to food when it reaches them in a certain distance and move away from ghosts when it reaches them in a certain distance as well. My agent had to react to food and ghosts in order to move and finish the game according to its rules. Therefore, I chose these two reflexive actions. I believe that, reciprocal (or negatives) of certain values is a good idea, because its computes faster, easy to handle and weights of the values helps the agent to detect the environments faster.

2. According to my test, I clearly see that AlphaBetaAgent test is faster than MinimaxAgent, since AlphaBetaAgent does pruning method to handle unnecessary branches -which are should not even be considered during the evaluation since certain branches will never get to reach upper nodes because of their values - of the minmax tree structure. I specifically observed this situation during solving of third question of the project.

3. Yes, it did, because the results of the minmax and alphabeta are the same however, second test did not iterate through the children of certain nodes and come back. Subsequently, AlphaBeta tree does not go through certain the nodes which are redundant nodes of the game. Therefore, both tests implemented same patterns with different speed.

4. According to my observations,AlphaBetaAgent is the fastest and minmaxAgent is faster than ExpectimaxAgent. Since, chance nodes might affect every node in the adversarial tree, we can not prune nodes from the tree. Therefore, naturally ExpectimaxAgent should be slower than AlphaBetaAgent. Subsequently, ExpectimaxAgent has more nodes (chance nodes) than minimaxAgent, therefore it iterates through more node in ExpectimaxAgent than minimaxAgent. Finally, we can say that AlphaBetaAgent is faster than minimaxAgent and minimaxAgent is faster than ExpectiMaxAgent.

5. I have not changed a lot my previous evaluation function to write better evaluation function. I additionally just used ghostScaredTimer to handle eating capsule and reacting to ghosts accordingly. I wrote that if ghostScaredTimer is bigger than zero, pacman will chase ghosts fast which gives it significant advantages to react.

6. I selected my features weights (food weight = .1, ghost weight =.1, scaredGhostweight = 1.0) significantly small, since if the weights were big, pacman would wait for them to make an action in a corner. However, when my features' weights are small, pacman immediately reacts to surroundings for example eating foods and running away from ghosts (or chasing ghosts, during the influence in power capsules. Using weights, makes coding significantly easier to handle how pacman behaves in certain conditions and also, its fast to compute (especially in small weights) and finally its good enough for default evaluation and better evaluation functions.