Hyunjin Lee 2020320023

Professor Hyunwoo Kim

COSE361-03

02 May 2021

Assignment #2

Development Environment: Visual Studio Code

1) Result:

```
Finished at 14:42:56

Provisional grades
==================
Question q1: 4/4
Question q2: 5/5
Question q3: 5/5
Question q4: 0/5
Question q5: 0/6
------------------
Total: 14/25
```

1) Code Explanations: (also commented in the python file)

Reflex Agent Code

I will consider following information: number of food left, distance between pacman's position and food position from the successor state, and distance between pacman's position and ghosts to prevent pacman from colliding with ghosts. If the following state has less food left than the current state, that indicates that the pacman has eaten a food and I will reward it with +10 score. If the pacman gets closer to a food in the successor state, I will also reward it with positive reciprocal minimum distance. If I simply consider the distance between pacman and the ghosts, pacman continuously shows the behavior of thrashing. Thus, I only consider when the distance between pacman and a ghost is below 2. If the distance gets below 2 and pacman has eaten a power pellet, I will give +100 score as the pacman gets closer to the ghost. On the other hand, if the distance gets below 2 and pacman has not eaten a power pellet, I will give -100 score because the action is extremely unfavorable.

Mini Max Agent Code

I have implemented two functions: maxValue and minValue. Both maxValue and minValue check whether the current state depth is equivalent to the depth provided and whether the current state is a lose/win state. If the condition is true, return the evaluated current state

value. maxValue function generates possible actions from current state and the successor states. Each successor state is entered into the minValue function as the next state must be the min state for the first ghost. The returned value is compared with the comparison variable v and v is returned after all the successor states have been looped through. On the other hand, minValue function keeps track of the agentIndex. The process is same as the maxValue function except for the fact that if the agentIndex is the last ghost's index, a maxValue function is called to evaluate pacman's maxValue states. If agentIndex is not the last ghost's index minValue function is called to evaluate the min value for the next ghost.

Alpha Beta Agent Code

The alpha beta code is similar to the mini max agent code. However, variables alpha and beta are considered both in maxValue and minValue functions. Alpha is initialized with -inf and beta is initialized with inf values, and the search continues down to the leaf or the depth node in the beginning. Alpha acts as a relative max value, and in the minValue function, if a given action's v value is smaller than alpha, v value is returned, and rest of the nodes are pruned (not considered). Beta acts as a relative min value, and in the maxValue function, if a given action's v value is greater than the beta value, v value is returned, and rest of the nodes are pruned as well.

2) Discussions:

1. Alpha-beta layers on top of minimax while cutting off large sections of the search tree. Alpha beta removes the subtrees that do not affect the final decision; by significantly reducing the number of static evaluations, the algorithm can explore up to twice as many levels as before. By simply limiting the search to more promising subtrees, we are thus able to explore the minimax tree more efficiently.
2. Reflex agents have very limited intelligence; accordingly, they require the environment to be fully observable. The only advantage reflex agents have over minimax and alpha beta is their simplicity. Thus, the reflex agent may perform better than the two algorithms if the goal state may be arrived quickly without interference of an opponent (ex. Ghosts) or in a non-zero-sum games.
3. Q: Branching factors vary with the game and the game state. How do we know how deep we go for the tree searches?
   A: Utilization of progressive deepening. If reaching the d level is concerning, provide an insurance policy by calculating the answer down to d – 1 level. To make sure we get an answer, we can use the progressive deepening from the first level and give ourselves an insurance policy for every level we calculate. This results in an answer at any time.