Pacman: Multi-Agent Search

1115201700113 Παναγιωτόπουλος Γεώργιος

Reflex Agent:

- I based my evaluation function on the premise that after a certain action that leads into a successor game state, the closer the food points are on average, the better the action. That means that my evaluation of an action that it leads to a state all the food points are relatevely close to the pacman should be bigger than my evaluation of a state which every food point is further away.
- For distance I used the Manhattan metric.
- As it turns out, summing all the inverses of the distances works suprisingly well (so much so that I used the same tactic for the better evaluation function) and yields far better results than subtracting by those distances or by their average.

Minimax:

- At the start of every call, I mod the agentIndex by the number of total agents. This operation allowed me to iterate through all the agents by simply increasing the agentIndex with every call, without the need for any extra checks.
- I increase the depth every time I am in max, since all the ghost actions take place in the same depth.
- The rest is basically adheres to the pseudocode that was presented in the lectures.

AlphaBetaAgent:

- Very similar to the minimax, basically copy-pasted and added an extra check in each case (min-max).

ExpectimaxAgent:

- The max portion the same as in minimax, the difference is that instead of min(), I sum the values of all the possible states that are generated by all the possible ghost actions.

Evaluation Function:

- Same as that of the reflex agent, the only difference being that now all the calculations are done for the gamestate itself rather than the successor gamestates of some actions.