Xinyi Ma [xim002@ucsd.edu](mailto:xim002@ucsd.edu)

Yijun Zhang [yiz160@ucsd.edu](mailto:yiz160@ucsd.edu)

Yuanchi Ha [yuha@ucsd.edu](mailto:yuha@ucsd.edu)

**HW4 Writeup**

**Algorithm:**

**P1: evaluationFunction**

**P2: MinimaxAgent**

In the minimax agent class, we defined function min\_value and max\_value separately. The recursion call is realized by a function called get\_value.

The get\_value function consider 2 cases. If the current index of the agent is 0, which means the current agent is Pacman, then we call the max\_value function. Otherwise, we call the min\_value function.

In min\_value function, we first define the value as infinite to update it with smaller value later. Then we consider two distinct conditions. If the game is won or lost already, we return the evaluationFunction(gameState) directly. Otherwise, we consider another two conditions. If the index of current agent is the same as the number of ghosts, which mean that we already evaluated all ghosts, then we call max\_value function with pacman’s index inside the loop; otherwise, we recursively call min\_value with next ghost’s index.

In max\_value function, the logic is smiliar, except that we define value as negative infinite to start with, so that we can update it with bigger value later. Moreover, after considering whether the game is win or lost, we don’t need to consider another two conditions. Instead, we call min\_value with the first ghost’s index to update the value inside the loop.

Outside these three functions, we have the main function to start the game. First, we get all the legal actions to loop with. We also define the result action as direction.stop and score as negative infinity to be updated later. Then we start the game by looping through all legal actions and update the scores each time by calling min\_value with the index of the first ghost inside inside a max. If the score is better than the previous one, then we update the result action with the new action too.

**P3: AlphaBetaAgent**

**P4: ExpectimaxAgent**

**P5: betterEvaluationFunction**

**Performance on test data:**

**P1: evaluationFunction**

Average Score: 1215.6

Scores: 1377.0, 1060.0, 1193.0, 1115.0, 1232.0, 1186.0, 1206.0, 1380.0, 1224.0, 1183.0

Win Rate: 10/10 (1.00)

Time: 4mins,42secs

**P2: MinimaxAgent**

Average Score: 84.0

Scores: 84.0

Win Rate: 0/1 (0.00)

Time: 55secs

**P3: AlphaBetaAgent**

Average Score: 84.0

Scores: 84.0

Win Rate: 0/1 (0.00)

Time: 57secs

**P4: ExpectimaxAgent**

Average Score: 84.0

Scores: 84.0

Win Rate: 0/1 (0.00)

Time: 56secs

**P5: betterEvaluationFunction**

Average Score: 1309.3

Scores: 1365.0, 1374.0, 1319.0, 1333.0, 1379.0, 1101.0, 1364.0, 1292.0, 1383.0, 1183.0

Win Rate: 10/10 (1.00)

Time: 2mins,24secs

**Analysis of result:**