**Udacity AI Nanodegree: Heuristic Analysis for Advanced Game Playing Agent Module Project**

**Omkar Vedpathak, April 2017 cohort**

**Custom scoring function 1**

The main custom scoring function uses one modification on top of the ‘improved score’ heuristic, whose goal is to maximize the difference between the player and the opponent moves. Boards in which the player has more moves relative to the opponent are favored, since this is likely to produce boards where the opponent runs out of moves, which leads to a win for the agent.

If there is ever a board state where the opponent has no legal moves remaining, and the agent has legal moves remaining, we should guarantee selection of this board state, since it is a winning board state. To guarantee selection of this board state, the utility is set to infinity. The downside of this improvement is that it is rarely actually triggered, since it requires large search depths (>7), since only after a decent number of plies would the opponent ever be in a state where they have no legal moves remaining. The opposite condition is also used to guide the heuristic, i.e. if the player is ever a loser, ensure that moves producing this condition are never selected, so we set the utility to negative infinity

The reason I chose this heuristic over the others, because the first enhancement should never be worse than the default improved score heuristic, since we are using the improved score heuristic plus knowledge that ensures that winning moves if detected are always selected, and losing moves if detected are always avoided. I also didn’t want to add unnecessary complexity as introduced by function 3, and additionally this function performed marginally better than the other scoring functions on 20 experimental test runs I performed using the tournament.py script (see table below). The time complexity of the implementation is also linear since we are only counting the number of possible moves of each player, which is always at most 8 per player, since the pieces can only move as Chess knights, and the number of legal moves left per player, to determine if either has won / lost.

**Custom scoring function 2**

This scoring function uses the default ‘improved\_score’ heuristic described above, divided by the total number of blank spaces on the board.

i.e. (P1 Moves – P2 Moves) / (number of blank spaces on the board)

This heuristic tries to maximize the numerator, i.e. the difference between P1 and P2 moves, and minimize the denominator, i.e. to produce board positions with fewer blank pieces. This could be a risky approach especially if the depth of search is low, since it may just produce situations in which there aren’t many spaces to move into, and it would tend to end games early.

**Custom scoring function 3**

This scoring method used in this function is the same as the ‘improved\_score’ heuristic in addition to a modification to take into account how far a move is away from the center of the board. This modification chooses moves that are more central. This is simply to try to favor moves where there are future moves, based on my intuition when attempting to play the game on a sample board. I took the square root of the center score, since it is a squared value by default, and added a scaling factor so its influence on the move utility is similar to the influence of number of moves. E.g. for a piece that is placed in the corner (0,0) at the beginning of the game, the (negative) utility contribution for this move is:

0.25 \* Sqrt((8/2-0)2 + (8/2-0)2)= 0.25 \* sqrt(32) = 0.25\* 5.65 = 1.41

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Match #** | **Opponent** | **AB\_Improved** | | **AB\_Custom** | | | **AB\_Custom\_2** | | | **AB\_Custom\_3** | | |
|  |  | **Won** | **Lost** | | **Won** | **Lost** | | **Won** | **Lost** | | **Won** | **Lost** |
| **1** | **Random** | 2 | 8 | | 6 | 4 | | 6 | 4 | | 1 | 9 |
| **2** | **MM\_Open** | 1 | 9 | | 1 | 9 | | 0 | 10 | | 1 | 9 |
| **3** | **MM\_Center** | 3 | 7 | | 2 | 8 | | 1 | 9 | | 5 | 5 |
| **4** | **MM\_Improved** | 1 | 9 | | 1 | 9 | | 1 | 9 | | 3 | 7 |
| **5** | **AB\_Open** | 7 | 3 | | 3 | 7 | | 6 | 4 | | 3 | 7 |
| **6** | **AB\_Center** | 5 | 5 | | 8 | 2 | | 6 | 4 | | 6 | 4 |
| **7** | **AB\_Improved** | 6 | 4 | | 6 | 4 | | 2 | 8 | | 7 | 3 |
|  | **Win Rate %** | 35.7 |  | | 38.6 |  | | 31.4 |  | | 37.1 |  |