Heuristic Analysis
Sashank Santhanam

In this project, I have evaluated 6 different heuristics and chosen the 3 best heuristics as my final 3 heuristics to evaluate.

```
This script evaluates the performance of the custom_score evaluation
function against a baseline agent using alpha-beta search and iterative
deepening (ID) called `AB_Improved`. The three `AB_Custom` agents use
ID and alpha-beta search with the custom_score functions defined in
game_agent.py.

                        **************************
                             Playing Matches
                        **************************

 Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                        Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1       Random      10  |   0    10  |   0    10  |   0     9  |   1
    2       MM_Open      7  |   3     9  |   1     7  |   3     9  |   1
    3      MM_Center     8  |   2    10  |   0     8  |   2     8  |   2
    4     MM_Improved    7  |   3     9  |   1     5  |   5     9  |   1
    5       AB_Open      6  |   4     3  |   7     7  |   3     6  |   4
    6      AB_Center     9  |   1     6  |   4     6  |   4     6  |   4
    7     AB_Improved    4  |   6     4  |   6     6  |   4     6  |   4
--------------------------------------------------------------------------
           Win Rate:      72.9%        72.9%        70.0%        75.7%
```

Figure 1: The first 3 heuristics

The above figure represents the results obtained from using the following heuristics.

1. AB_Custom - #my_moves – 2*#opponent_moves. This is an aggressive heuristic which makes the agent chase after the opponent.

2. AB_Custom2- #my_moves/(#opponent_moves+1). This is a defensive heuristic.

3. AB_Custom3 100* my_moves – 50 * opponent_moves . So this heuristics is basically a reversal of the first heuristic with a multiplier. The heuristic could be returned as 50(2*my moves- opponent_moves) .

Using the above 3 heuristics, we were able to obtain a lower bound of 70% and achieved an higher heuristic of 75.7% which is a good improvement and also beats the AB_Improved by 2.5% which is a good metric.

The next 3 heuristics tried out were based out on the position of the player and the opponent.

4. The first heuristic in this method was to calculate the manhattan distance between the player and the opponent.

5. The second heuristic was to calculate the Euclidean distance between the player and the opponent.

6. The third heuristic used the process of taking the difference of the distance between the players calculated from the center.

```
This script evaluates the performance of the custom_score evaluation
function against a baseline agent using alpha-beta search and iterative
deepening (ID) called `AB_Improved`. The three `AB_Custom` agents use
ID and alpha-beta search with the custom_score functions defined in
game_agent.py.

                        **************************
                             Playing Matches
                        **************************

 Match #    Opponent     AB_Improved    AB_Custom    AB_Custom_2  AB_Custom_3
                          Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1        Random        8  |   2     10  |   0     10  |   0     9  |   1
    2        MM_Open       5  |   5      6  |   4      7  |   3     8  |   2
    3       MM_Center      8  |   2     10  |   0      8  |   2     9  |   1
    4     MM_Improved      7  |   3      8  |   2      7  |   3     8  |   2
    5        AB_Open       6  |   4      4  |   6      3  |   7     2  |   8
    6       AB_Center      9  |   1      6  |   4      4  |   6     7  |   3
    7     AB_Improved      6  |   4      5  |   5      2  |   8     3  |   7
--------------------------------------------------------------------------
           Win Rate:      70.0%        70.0%        58.6%        65.7%
```

The best performing heuristic was based on the Manhattan distance between the 3 heuristics.

Comparing the 6 different heuristics, the top 3 heuristics that I found were all based on the number of moves available for a player without considering the player location.

Best Heuristic:- The best heuristic would be 50(2*my moves- opponent_moves). The main reason for choosing this heuristic as the best one is the comparison of its performance with its peer's heuristics. This heuristic was able to achieve a win rate of 75.7% which is significantly better than all the other heuristics I had evaluated. Another reason for choosing this heuristic is its performance advantage over AB_Improved.