Heuristic Analysis

My Custom Heuristic functions

In order to win the game, there are two specific ways that a player has to follow. Firstly, the offensive behavior consists trying to obstruct the opponent's game, while the defensive behavior consists trying to keep as many options as possible.

Custom_score_1

It is same as improved_score except it return the own moves divided by the opponent moves. The reasoning behind this is I want my agent to play more efficiently. I might want to sacrifice some of my own moves to dynamically decrease my opponent's move. For example, there are two moves lead to either (my_moves = 10,opp_move = 5) or (my_move = 4, opp_move = 1). In the case of Improved_score, it would choose (my_moves = 10,opp_move = 5) but my function will choose the latter one and it's more reasonable to do so.

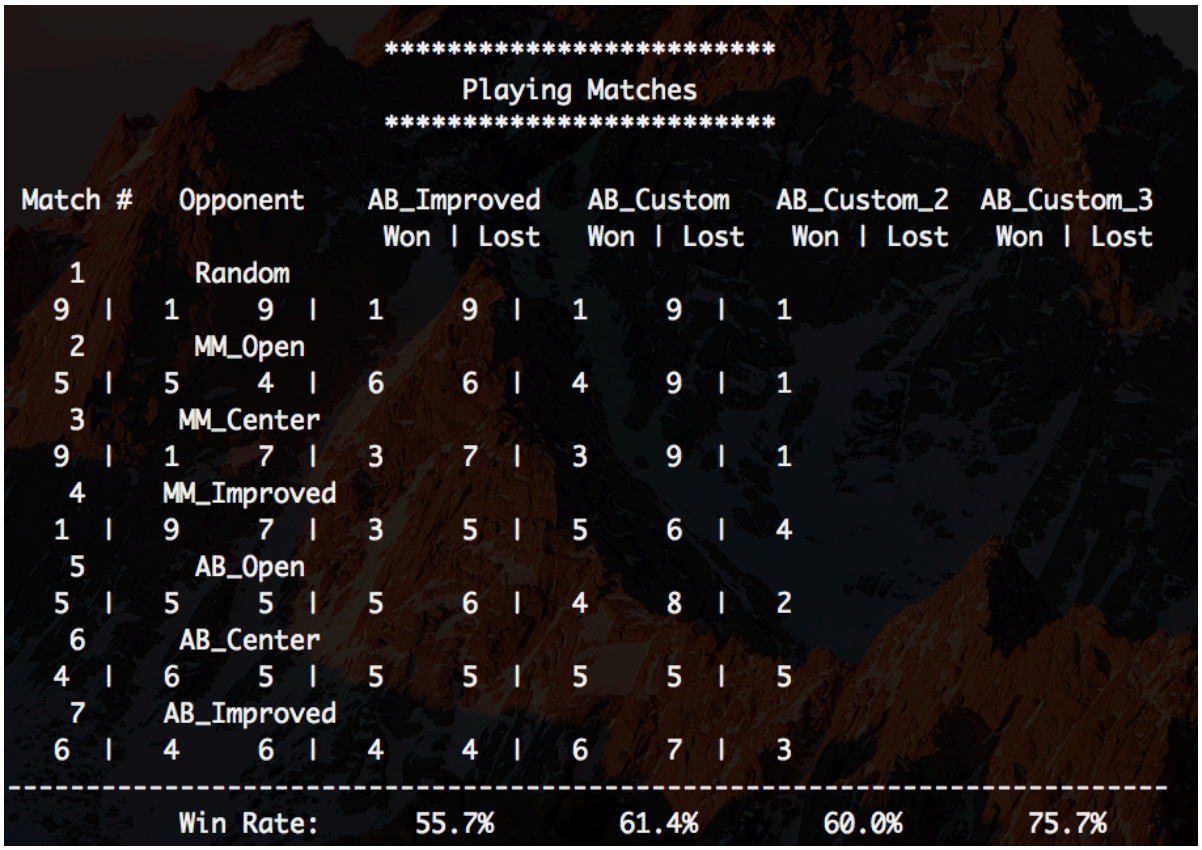
Custom_score_2

The same as improved_score except it has a different way to calculate the final score. Most of the time, the agent forced itself to escape to the edge or corner, and it will be easily trapped on that position. We want the agent smartly chose a move not only result in more "next moves", but also a defensive position where opponent can't easily trap you. This image shows the black knight is trapped at the edge. Therefore in my calculation, I reduce the weight on the edge moves and corner moves.

Custom_score_3

A combination of Custom_score_1 and Custom_score_2. The function uses the better weight on edge move or corner move. In this case, it returns my score divided by opponent score.

Results



```
**************************
    Playing Matches
**************************

Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                       Won | Lost   Won | Lost   Won | Lost   Won | Lost
   1       Random
  9  |   1     9  |   1     9  |   1     9  |   1
   2       MM_Open
  5  |   5     4  |   6     6  |   4     9  |   1
   3      MM_Center
  9  |   1     7  |   3     7  |   3     9  |   1
   4     MM_Improved
  1  |   9     7  |   3     5  |   5     6  |   4
   5       AB_Open
  5  |   5     5  |   5     6  |   4     8  |   2
   6      AB_Center
  4  |   6     5  |   5     5  |   5     5  |   5
   7     AB_Improved
  6  |   4     6  |   4     4  |   6     7  |   3
----------------------------------------------------------------------------
          Win Rate:      55.7%        61.4%        60.0%        75.7%
```

The performance of agents is showing in the image above, which is the data results form tournament.py. From the data, we really can't tell much as they are doing generally badly except the AB_custom_score. However, this is none iterative verses Ture alphabeta iterative, and probably my computer is powerful, so it's not fair at all.

Base on the performance of the three heuristics, the third heuristic has higher win rate than others, which means that the path towards an optimal heuristic is the combination of Custom_score_1 and Custom_score_2. Also, it proved that in the better-weighted sum of all elements that contribute to evaluating move rather than an overly aggressive or defensive behavior. In addition, the third heuristic is the only one that did not lose in all of the seven different testing. Therefore, the third heuristic function is the recommendation.