Jovani Rico
July 27, 2017

Heuristic Analysis

The following image is the result obtained by the agents.

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

Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                     Won | Lost   Won | Lost   Won | Lost   Won | Lost
   1       Random     7  |  3     10  |  0     9  |  1     9  |  1
   2       MM_Open    6  |  4     10  |  0     6  |  4     9  |  1
   3      MM_Center   8  |  2      9  |  1    10  |  0     8  |  2
   4     MM_Improved  8  |  2      8  |  2     7  |  3     8  |  2
   5       AB_Open    5  |  5      5  |  5     6  |  4     7  |  3
   6      AB_Center   4  |  6      6  |  4     6  |  4     5  |  5
   7     AB_Improved  2  |  8      4  |  6     3  |  7     5  |  5
-----------------------------------------------------------------------
          Win Rate:      57.1%        74.3%        67.1%        72.9%
```

AB_Custom

The idea of the heuristic was to limit the number of moves of the opponent. As shown in Figure, the performance of this agent was slightly above of the AB_Improved agent with 74.3%. The custom heuristic was implemented in order to chase the opponent as follow:

#my_moves - 2 * #opponent_moves

Code:

```
return float( len ( game.get_legal_moves( player ) ) - 2 *
len( game.get_legal_moves( game.get_opponent(player) ) ) )
```

AB_Custom_2

This heuristic is a mix of defense and offense, but paying attention to the parameters as the game advances. Towards the end of the game, the number of common moves should increase in importance and the number of moves available should decrease. This heuristic algorithm got an improvement of 67.1%.

Code:

```
    opp = game.get_opponent(player)
    opp_moves = game.get_legal_moves(opp)
    my_moves = game.get_legal_moves()
    common_moves = opp_moves and my_moves
    ratio = 1 / (game.move_count + 1)
    return float(len(common_moves) * ratio + (game.move_count + 1) *
len(game.get_legal_moves()))
```

AB_Custom_3

This heuristic algorithm is a combination of two logics, the first one is that player should have more moves in comparison to opponent and the second logic is that opponent should have less moves in comparison to player. Mixing these two, we got a better performance of the AB_Improved agent with 72.9%. We can express this as follow:

$$(len(\text{my\_moves})) \wedge 2 - \beta (len(available\ opponent\ moves)) \wedge 2$$

Code:

```
    my_moves = len(game.get_legal_moves(player))
    opponent_moves = len(game.get_legal_moves(game.get_opponent(player)))
    return my_moves * my_moves - 1.5 * opponent_moves * opponent_moves
```

β was chosen empirically.