Heuristic Analysis

AIND-Isolation

Heuristic 1

Improved Score

own_moves = Moves of first player
opp_moves = Moves of opponent player

```
if game.is_loser(player):
    return float("-inf")

if game.is_winner(player):
    return float("inf")

own_moves = len(game.get_legal_moves(player))
opp_moves = len(game.get_legal_moves(game.get_opponent(player)))
return float(own_moves - opp_moves)
```

This is the heuristic given in the lectures. It outputs "a score equal to the difference in the number of moves available to the two players". As long as the number returned is positive, we will have a better chance of winning.

Heuristic 2

Custom Score 1

opponent = opponent player
my_moves = moves of the first player
opponent_moves = moves of the opponent player

```
opponent = game.get_opponent(player)
my_moves = len(game.get_legal_moves(player))
opponent_moves = len(game.get_legal_moves(opponent))

h1 = float(my_moves - 2*opponent_moves)
```

This heuristic is the one mentioned in the lectures. It is meant to be a more aggressive strategy, by giving more weight to the opponent moves, so we are always chasing the opponent.

Heuristic 3

Custom Score 2

opponent = opponent player
my_moves = moves of the first player
opponent_moves = moves of the opponent player
agg_factor = factor that will help us lower or increase aggressiveness

```
opponent = game.get_opponent(player)
my_moves = len(game.get_legal_moves(player))
opponent_moves = len(game.get_legal_moves(opponent))
agg_factor = 1

if my_moves-opponent_moves<=0:
   agg_factor = 3
else:
   agg_factor = 1

h2 = float(my_moves - agg_factor*opponent_moves)
```

This heuristic is meant to balance aggressiveness. If we are loosing, we will be more aggressive by incrementing opponent moves by 3. If we are winning, we will keep playing normally

Heuristic 4

Custom Score 3

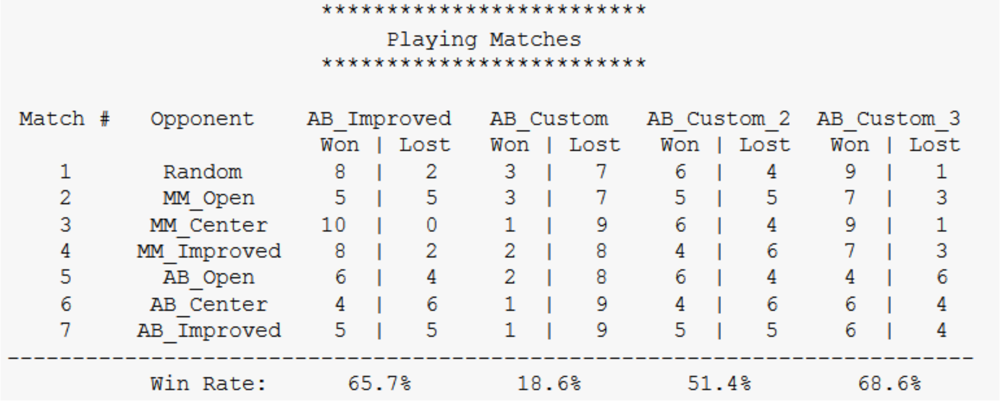
opponent = opponent player
my_moves = moves of the first player
opponent_moves = moves of the opponent player
height = center position
width = center position
posx = player location on x axis
posy = player location on y axis
center = center of the first player
center_opp = center of the opponent player

```
 opponent = game.get_opponent(player)

height = game.height/2
width  = game.width/2

posx, posy = game.get_player_location(player)
posx_o, posy_o = game.get_player_location(opponent)

center = abs(height-posy)+abs(width-posx)
center_opp =  abs(height-posy_o)+abs(width-posx_o)
my_moves = len(game.get_legal_moves(player))
opponent_moves = len(game.get_legal_moves(opponent))

h3 = float((center*my_moves)-(center_opp*opponent_moves))
```

This heuristic is taking into account how far each of the players are from the center. If we are near the center, we will play aggressively

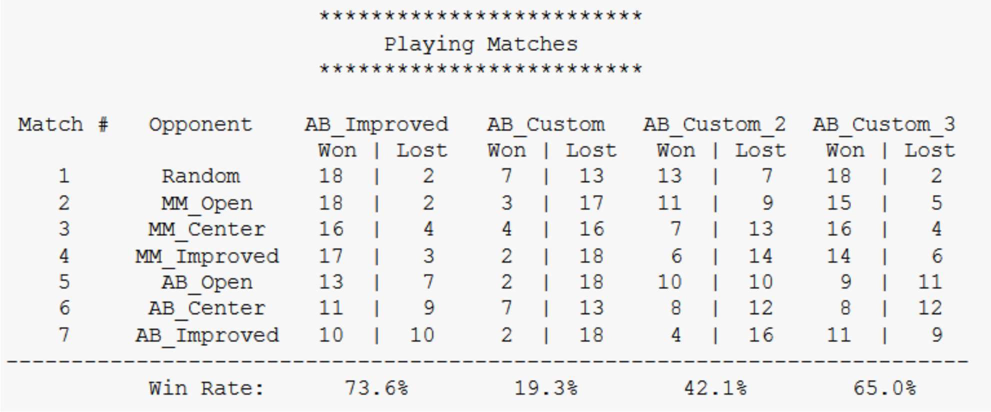
Results

First run:



```
                         *************************
                              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     3  |   7     6  |   4     9  |   1
    2       MM_Open      5  |   5     3  |   7     5  |   5     7  |   3
    3      MM_Center    10  |   0     1  |   9     6  |   4     9  |   1
    4     MM_Improved    8  |   2     2  |   8     4  |   6     7  |   3
    5       AB_Open      6  |   4     2  |   8     6  |   4     4  |   6
    6      AB_Center     4  |   6     1  |   9     4  |   6     6  |   4
    7     AB_Improved    5  |   5     1  |   9     5  |   5     6  |   4
--------------------------------------------------------------------------
           Win Rate:      65.7%        18.6%        51.4%        68.6%
```

Second run:



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

 Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                        Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1       Random      18  |   2     7  |  13    13  |   7    18  |   2
    2       MM_Open     18  |   2     3  |  17    11  |   9    15  |   5
    3      MM_Center    16  |   4     4  |  16     7  |  13    16  |   4
    4     MM_Improved   17  |   3     2  |  18     6  |  14    14  |   6
    5       AB_Open     13  |   7     2  |  18    10  |  10     9  |  11
    6      AB_Center    11  |   9     7  |  13     8  |  12     8  |  12
    7     AB_Improved   10  |  10     2  |  18     4  |  16    11  |   9
--------------------------------------------------------------------------
           Win Rate:      73.6%        19.3%        42.1%        65.0%
```

Comparing each of the custom heuristics to the AB_Improved, we got:

- Being aggressive the whole time doesn't benefit us at all, we lose most of the time.
- Being aggressive just in certain parts of the game, is a better approach, but it still doesn't get us to the result of just calculating remaining moves from AB_Improved
- The third approach, and the one recommended, is the one that takes into account how far are the players in the board, playing near the center get us a much better result, as shown by the tests, we got close to the AB_Improved and even got a better result in the first run.