# Heuristic Analysis

The three heuristics I experimented used the number of moves the player had, the number of moves the opponent had, and the player location in the board in each ply.

The heuristic custom_score uses the number of legal player moves, and the location of the player. It gives importance to the number of legal moves, but it also penalizes moves that move away from the center of the board. The penalization will be the same either the player moves away in the x axis or in the y axis, so if it moves away in both axis it gives the double of the penalization. This will cause the agent to try to occupy the center of the board and then move away from the center in a uniform manner, but always giving importance to the number of plays available to him. This heuristic got a winning rate of 75,1 % , the only heuristic with a winning rate superior to that of the baseline.

```
own = len(game.get_legal_moves(player))
h, w = game.get_player_location(player)
# centrality will be bigger the more the player is located in the center of the board
centrality = abs( (game.height/2) - h) + abs( (game.width/2) - w)
# the heuristic will give more importance to the number of player moves, but it will
# penalize plays that go further away from the center of the board
heuristic = 2 * own - centrality
return float(heuristic)
```

*1 - custom_score*

The heuristic custom_score_2 uses only the location of the player. It gives better results the closer the player is to the edges of the board. Since it gives the same reward to either edge, when the player is in a corner it will give maximum score, and when the player is in the center of the board it will receive the least score possible. This heuristic got a winning rate of 70,6% .

```
h, w = game.get_player_location(player)
# selects the vertical axis location of the player, counting from the side where
# the player is closer
y_axis = min( abs(game.height - h), abs(h - game.height) )
# selects the horizontal axis location of the player, counting from the side where
# the player is closer
x_axis = min( abs(game.width - w), abs(w - game.width) )
# this heuristic tries to minimize the x_axis and y_axis measures, this way the agent will try to maintain near the edges
heuristic = (game.height - y_axis) + (game.width - x_axis)
return float(heuristic)
```

*2 - custom_score2*

The heuristic custom_score_3 uses the number of legal player moves minus 2 times the number of opponent moves. This heuristic makes the agent chase after the opponent, because it gives more importance to minimize the opponent's moves than to maximizing its own (given the choice of minimizing the opponent's moves by 2 or maximizing its own moves by 2 this agent will choose the first alternative). This gives the impression the agent is chasing after the opponent. This heuristic got a winning rate of 72,9 % .

```
opp_moves = len(game.get_legal_moves(game.get_opponent(player)))
own = len(game.get_legal_moves(player))
heuristic = own - 2 * opp_moves
return float(heuristic)
```

*3 - custom_score3*

The tournament.py script was run with a number of matches = 25 , five times superior to the original number of matches, in order to reduce randomness.

| Match# | Opponent | AB_Improved | | AB_Custom | | AB_Custom2 | | AB_Custom3 | |
|---|---|---|---|---|---|---|---|---|---|
| | | Won | Lost | Won | Lost | Won | Lost | Won | Lost |
| 1 | Random | 50 | 0 | 48 | 2 | 49 | 1 | 49 | 1 |
| 2 | MM_Open | 45 | 5 | 43 | 7 | 39 | 11 | 46 | 4 |
| 3 | MM_Center | 48 | 2 | 49 | 1 | 48 | 2 | 50 | 0 |
| 4 | MM_Improved | 39 | 11 | 41 | 9 | 38 | 12 | 39 | 11 |
| 5 | AB_Open | 26 | 24 | 25 | 25 | 27 | 23 | 23 | 27 |
| 6 | AB_Center | 31 | 19 | 29 | 21 | 24 | 26 | 27 | 23 |
| 7 | AB_Improved | 22 | 28 | 28 | 22 | 22 | 28 | 21 | 29 |
| Winning Percentage | | 74,6% | | 75,1% | | 70,6% | | 72,9% | |

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

Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                       Won | Lost   Won | Lost   Won | Lost   Won | Lost
   1       Random      50  |   0    48  |   2    49  |   1    49  |   1
   2       MM_Open     45  |   5    43  |   7    39  |  11    46  |   4
   3      MM_Center    48  |   2    49  |   1    48  |   2    50  |   0
   4     MM_Improved   39  |  11    41  |   9    38  |  12    39  |  11
   5       AB_Open     26  |  24    25  |  25    27  |  23    23  |  27
   6      AB_Center    31  |  19    29  |  21    24  |  26    27  |  23
   7     AB_Improved   22  |  28    28  |  22    22  |  28    21  |  29
--------------------------------------------------------------------------
          Win Rate:      74.6%       75.1%       70.6%       72.9%
```

With these results we can conclude that between the 3 heuristics used, the better one is the first one, which tried to own the center of the board while paying attention to the number of legal moves. The second heuristic is worse than the first one, either because it doesn't have into account the number of legal moves, or because owning the edges is a worse strategy than owning the center, or both. The third heuristic only has into account the number of legal moves, and in my previous tests I checked this aggressive heuristic has better results than others that only use this variable. Even so, it is worse than the first heuristic, making me believe that the best possible heuristic would have into account the number of legal moves and the location of the player.