```
**************************
     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     9  |   1     7  |   3     9  |   1
    2       MM_Open      5  |   5     8  |   2     5  |   5     6  |   4
    3      MM_Center     8  |   2     8  |   2     9  |   1     9  |   1
    4     MM_Improved    8  |   2     9  |   1     5  |   5     6  |   4
    5       AB_Open      7  |   3     6  |   4     5  |   5     5  |   5
    6      AB_Center     5  |   5     6  |   4     5  |   5     7  |   3
    7     AB_Improved    5  |   5     5  |   5     3  |   7     3  |   7
----------------------------------------------------------------------------
           Win Rate:      65.7%        72.9%        55.7%        64.3%
```

Heuristic 1
---

my_moves – (1.8) * opponent_moves

This heuristic selects the move which will take the player closer to the opponent. Such that during the endgames, there will be very less chances left with the opponent. The value 1.8 is selected after experimenting.

Heuristic 2

(my_moves – opponent_moves)/total_moves

This heuristic checks the accessibility of the player to the empty cells on the board. Higher the accessibility, higher the chance of win

Heuristic 3

This heuristic selects the adjacent node with minimum degree as the next cell. (Based on Warnsdorff's heuristic for Knight's tour problem)

Recommendation

Heuristic 1 is performing better in this case.

- It's easy to implement
- It's a constant time heuristics
- Out of 5 samples, heuristics 1 performed most consistently and better (>70%) all of the time.