Result Analysis of the custom heuristics function

In this project we have experimented with three different heuristic functions, these three functions are the following:

- Custom_score(), in this heuristic I made the same calculation as in heuristic 2 but if the result of the difference is 0 I return the difference of the distances of both player to the center calculated with the Manhattan distance.
- Custom_score_2(), this heuristic return simply the difference between the legal moves of my player and the legal moves of the opponent.
- Custom_score_3(), this heuristic return the difference of legal moves between my player and the opponent and take in count the distance of my move from the center of the gameboard.

Here is a summary of my 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	64 | 16	62 | 18	55 | 25	65 | 15
2	MM_Open	60 | 20	48 | 32	45 | 35	51 | 29
3	MM_Center	57 | 23	60 | 20	45 | 35	47 | 33
4	MM_Improved	30 | 50	26 | 54	27 | 53	27 | 53
5	AB_Open	40 | 40	47 | 33	41 | 39	40 | 40
6	AB_Center	47 | 33	41 | 39	44 | 36	38 | 42
7	AB_Improved	48 | 32	37 | 43	32 | 48	40 | 40
	Win Rate:	61.8%	57.3%	51.6%	55.0%

The best performance we had is with the Custom_Score heuristic but very near form the Custom_Score_3 heuristic. I changed the NUM_MATCHES parameter setting it to have 80 plays for each round in order to have a better understanding of the performance of the functions.

Below we provide a description of each function and how each perform against every type of opponent provide by Udacity.

Heuristic 1 – Implemented in function Custom_score()

With this heuristic function, we return the difference in number of available legal moves left between the players adding the distance of the move to the center of the board. So the evaluation value take in count and corrects the difference. If the move is near the center will add a higher value than is far. If the returned value is "inf" ("-inf"), then `player` has won (lost) the game.

This heuristic is the best that performs between the three I implemented. It benefits from positional advantage better than in the third case.

Heuristic 2 – Implemented in function Custom_score_2()

With this heuristic function, we return the difference in number of available legal moves left between the players. If both players have the same number of moves, then the returned value is zero. If the returned value is positive (negative), then the student player is doing better (worse) than its opponent. If the returned value is "inf" ("-inf"), then the student has won (lost) the game.

The performance of this function is not very good. It has some benefits that are the **easily** interpretable and fast to compute, but on the other hand it doesn't take in count the position of the student player and the opponent

Heuristic 3 – Implemented in function Custom_score_3()

With this heuristic function, we return the difference in number of available legal moves left between the players. If both players have the same number of moves, then the returned value is the scale difference between the Manhattan distance of each player to the center of the board. If the returned value is positive (negative), then the student player is doing better (worse) than its opponent. If the returned value is "inf" ("-inf"), then the student has won (lost) the game.

This heuristic performs a bit better than the heuristics 2, but worse than the first explained. It benefits from positional advantage, but not as in the first case.