1 Heurestic function analysis

The last heuristic is the weighted heuristic function. one it takes into account the number of available moves for each player, and evaluates which one has more options:

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


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


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


    return 1.3*my_moves**2- opponent_moves**2
```

The second heuristic takes into account the fact if the player and opponent have common moves that they can make. As the game progresses, it put more emphasis on the common moves rather than available moves. It is defined as follows:

```
def custom_score_2(game, player):
my_moves = game.get_legal_moves(player)
    opponent_moves = game.get_legal_moves(game.get_opponent(player))
    common_moves = opponent_moves and my_moves
    if not opponent_moves:
        return float("inf")
    if not my_moves:
        return float("-inf")
    return float(len(common_moves) *
    1 / (game.move_count + 1) + (game.move_count + 1) *
    len(game.get_legal_moves()))
```

Chosen heuristic is based on the centrality of the players position. By playing this strategy, players have higher number of options available further. The closer is the player to the board, the higher are the chances of him losing.

```
def custom_score_3(game, player):
    if game.is_winner(player):
        return float("inf")
    if game.is_loser(player):
        return float("-inf")

    center = game.width/2
    my_moves = len(game.get_legal_moves(player))
    opponent_moves = len(game.get_legal_moves(game.get_opponent(player)))

    my_position = game.get_player_location(player)
    opponent_position = game.get_player_location(game.get_opponent(player))

    return float(my_moves - opponent_moves +
    abs(center - my_position[0])+
    abs(center + my_position[1]) -
    abs(center - opponent_position[0]) -
    abs(center - opponent_position[1]))
```

The results of the tournament with heuristic functions against the opponents are:

****

Playing Matches

****

Match #	Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3	
		Won	Lost	Won	Lost	Won	Lost	Won	Lost
1	Random	9	1	10	0	10	0	10	0

2	MM_Open	10	0	9	1	10	0	7	3
3	MM_Center	9	1	10	0	10	0	10	0
4	MM_Improved	8	2	9	1	8	2	9	1
5	AB_Open	8	2	5	5	5	5	6	4
6	AB_Center	6	4	5	5	2	8	5	5
7	AB_Improved	5	5	6	4	6	4	5	5
	Win Rate:	78.6%		77.1%		72.9%		74.3%	

As we can see the weighted heuristic with iterative deeping provides the best winning results, beating the opponents in 78.6 % of cases. The depth limited custom score provides the results not so different. The next in performance is the heuristic function 3 with 74.3% won cases and the last is the custom function 2 with 72.9% winning cases