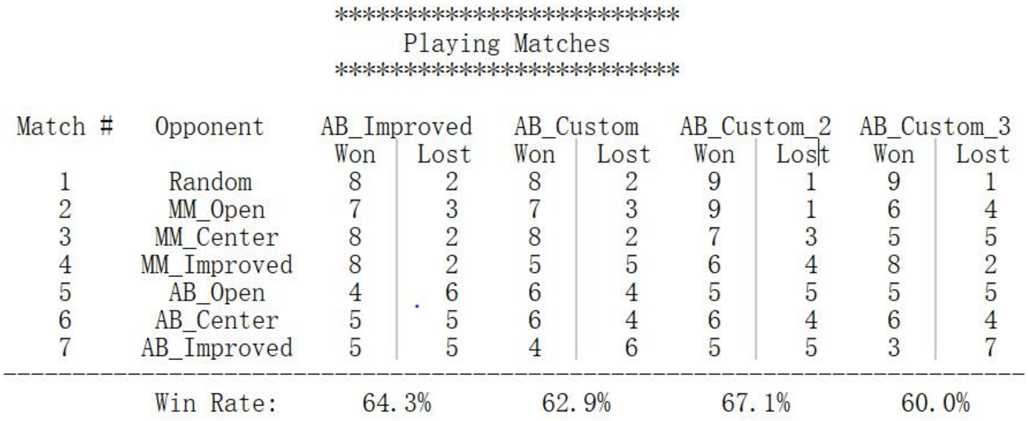
Analysis Of The Heuristics Result

I has implemented three kind of evaluation heuristics.The first only change the opponent players params which act as the improved_score heuristics. The second combines the improved_score and center_score, considering that influence of the distance to the center. The last one consider the average moves of my next steps that I can has as where as the second one's influence.

So the result shows below:



```
                     *************************
                          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     8  |   2     9  |   1     9  |   1
    2       MM_Open      7  |   3     7  |   3     9  |   1     6  |   4
    3      MM_Center     8  |   2     8  |   2     7  |   3     5  |   5
    4     MM_Improved    8  |   2     5  |   5     6  |   4     8  |   2
    5       AB_Open      4  |   6     6  |   4     5  |   5     5  |   5
    6      AB_Center     5  |   5     6  |   4     6  |   4     6  |   4
    7     AB_Improved    5  |   5     4  |   6     5  |   5     3  |   7
--------------------------------------------------------------------------
           Win Rate:      64.3%        62.9%        67.1%        60.0%
```

Figure: the performance of my three evaluation heuristics

We win more games when opponents only use minmax or random method. The AB_Custom_2 perform best in this round_robin games. On the whole, it is not easy to win when all of us using alpha_beta strategy.

As we can see, the original AB_Improved act good against pre-defined agents(yes, I tried several times, It performs almost one of the best in each round). So I think ,improved_score heuristic is a good function in this kind of game, because:

1. The rule suggests that we must improve our move space and decrease the opponent's space.
2. It consider two side state of two opponents.
3. It has stable performance.

The open_move_score is not shown in the figure. It performs general, But not better than the improved_score.

When we considering that decreasing the space of opponent's move space, we increase the weights so that it has a bigger influence on the score:

```
return float(len(game.get_legal_moves(player))
                    - 1.2 * len(game.get_legal_moves(game.get_opponent(player))))
```

But sometimes it perform better than the default one, sometimes not. So think this kind of method may has some influence not obviously.

The third perform a litter better when i considering the distance to the center of the location:

```
return custom_score(game, player) - abs(r - 2) / float(4)
```

I think that It has more moving space when the location not in the center and not at the edge.
So I minus 2 then get the abs of the value. I think it perform best because:

1. This heuristic method considers current state of the legal moves both sides.
2. It consider the distance to the center of player1 location
3. It combines the effect together

But it also depends. Sometimes when changing the parameters it will not perform that better like this one.

At last, I introduce the concept that my move's score will be influenced by the moving space of next step. I make the method likes below:

```
sum(next_moves) / (8 * float(len(next_moves))) + custom_score_2(game, player)
```

It calculates the average moving counter next after all moving possibilities. But as shown in the figure. It helps noting at all(After my several trials, it does have little help).

After all, I will choose the improved_score, It performs better than others after trials. But my heuristics can not defeat the ID_Improved. So sad.