Heuristic Analysis

Implement and Analyze

I‘ve got an inspiration from this chessprogramming wesite (http://chessprogramming.wikispaces.com/Evaluation).
The heuristics function can be a linear summation of several factors and what I can do is to pick up some features. Compare them with each other. If needed, combine them and fine-tune the weights of them. So I pick up three features of them

1. My relative mobility: my moves relative to opponent's moves - inspired by improved_score function from sample_players.py
2. My relative center mobility: my moves distance to center relative to opponent's move distance to center, staying around the center gives the player more chances not to be partitioned and trapped - inspired by center_score function from sample_players.py
3. Center Mobility: occuppied center or not

The former two features are divided by 'my_value' in order to dimensionless, so they could be tuned by weights.

Here are how these three function are implemented:

- custom_score:
  center_mobility + my_relative_mobility + my_relative_center_mobility
- custom_score_2:
  my_relative_mobility
- custom_score_3:
  my_relative_center_mobility

My purpose here is to find out which factor is better and combine them to get a better performance.

Performance

Playing Matches(NUM_MATCHES = 50)

Match #	Opponent	AB_Improved	AB_Custom	AB_Custom_2	AB_Custom_3
		Won - Lost	Won - Lost	Won - Lost	Won - Lost
1	Random	85 - 15	80 - 20	76 - 24	78 - 22
2	MM_Open	68 - 32	58 - 42	66 - 34	53 - 47
3	MM_Center	77 - 23	81 - 19	67 - 33	71 - 29
4	MM_Improved	61 - 39	61 - 39	54 - 46	52 - 48
5	AB_Open	55 - 45	49 - 51	47 - 53	45 - 55
6	AB_Center	60 - 40	52 - 48	56 - 44	46 - 54
7	AB_Improved	51 - 49	47 - 53	44 - 56	39 - 61
	Win Rate:	65.3%	61.1%	58.6%	54.9%

As you can see above, under a condition of "NUM_MATCHES = 50", compared to these three custom functions, the winning rate is decreasing. Compared Custom_2 with Custom_3, it shows that the result of my_relative_mobility factor is better than my_relative_center_mobility, because the former factor refers to the opponent's move rather than only staying as close as to the center. Custom is the best among theses three, because it takes into account all these factors.

Opponent	**AB_Custom**	**AB_Custom_2**	**AB_Custom_3**
mini_max	much better	better	better
alpha_beta	almost equal	slightly worse	much worse
AB_Improved	slightly worse	worse	much worse

Looking into the opponents in detail, all custom functions are better than the mini-max opponents due to the efficient search of alpha-beta iterative deepening. However, compared to alpha-beta opponens themselves, they are slightly worse because the custom functions contain more time cost calculation steps.
Compared to AB_Custom with AB_Improved, AB_Custom is slightly worse than AB_Improved also because the calculation is more time cost than AB_Improved.

Recommendation

According to the winning rate, I recommend that we should use AB_Improved.

1. Its calculation is simple and time saving only considering the difference between my moves and opponent moves.
2. The calculation considers the difference between my moves and opponent moves which maximizes my moves in the same time trying to reduce the mobility of opponent mvoes.
3. According the comparison between my custom_2 and custom_3, it shows that my moves relative to opponent moves dominate the influence factor in the game. By only using the most influential factor, the evaluation will be simple but powerful.