

Heuristic Analysis:

The three heuristic functions I choose are as below.

- (1) The first heuristic function I tried is that in the beginning of the game (when # of blanks are more than half of total cells), use $\#my_move - 2 * \#oppo_move$, after that I use $\#my_move - \#oppo_move$.

I was thinking that when the game just started, we could use a more aggressive method. As game went on, the number of available moves decreased, so it's possible that the $\#my_move - 2 * \#oppo_move$ became negative while $\#my_move - \#oppo_move$ was still positive so that agent would make a better choice using $\#my_move - \#oppo_move$.

Below is one of the test results. Seems this function did not work well as I expected.

```
*****
Evaluating: ID_Improved
*****

Playing Matches:
Match 1: ID_Improved vs Random      Result: 18 to 2
Match 2: ID_Improved vs MM_Null     Result: 17 to 3
Match 3: ID_Improved vs MM_Open     Result: 15 to 5
Match 4: ID_Improved vs MM_Improved Result: 17 to 3
Match 5: ID_Improved vs AB_Null     Result: 16 to 4
Match 6: ID_Improved vs AB_Open     Result: 9 to 11
Match 7: ID_Improved vs AB_Improved Result: 10 to 10

Results:
ID_Improved      72.86%

*****
Evaluating: Student
*****

Playing Matches:
Match 1: Student vs Random      Result: 18 to 2
Match 2: Student vs MM_Null     Result: 13 to 7
Match 3: Student vs MM_Open     Result: 18 to 2
Match 4: Student vs MM_Improved Result: 15 to 5
Match 5: Student vs AB_Null     Result: 14 to 6
Match 6: Student vs AB_Open     Result: 10 to 10
Match 7: Student vs AB_Improved Result: 12 to 8

Results:
Student          71.43%
```

(2) The second heuristic function I tried is $\#my_moves / \#oppo_moves$. I replace the subtraction in the function " $\#my_moves - \#oppo_moves$ " with division. Assume we have below two choices for the next move, Move choice 1: $\#my_moves = 4$, $\#oppo_moves = 2$ and Move choice 2: $\#my_moves = 3$, $\#oppo_moves = 1$. The function $\#my_move - \#oppo_move$ gives a tie between the two choices, so computer might pick choice 1, however, the best move should be choice 2 as there will be only 1 move left for opponent if choosing this move. Using $\#my_moves / \#oppo_moves$ yields this result.

I think the idea behind this is that $\#my_moves / \#oppo_moves$ is that due the characteristic of the the $y=1/x$ curve, when maximizing the value of this division, the $\#oppo_moves$ will be greatly reduced, much faster than the linear function of $\#my_moves - \#oppo_moves$. Below is one of the test results with $NUM_MATCHES = 25$

```
*****
Evaluating: ID_Improved
*****

Playing Matches:

Match 1: ID_Improved vs Random      Result: 86 to 14
Match 2: ID_Improved vs MM_Null     Result: 81 to 19
Match 3: ID_Improved vs MM_Open     Result: 81 to 19
Match 4: ID_Improved vs MM_Improved Result: 76 to 24
Match 5: ID_Improved vs AB_Null     Result: 71 to 29
Match 6: ID_Improved vs AB_Open     Result: 51 to 49
Match 7: ID_Improved vs AB_Improved Result: 55 to 45
```

```
Results:
ID_Improved      71.57%
```

```
*****
Evaluating: Student
*****
```

```
Playing Matches:

Match 1: Student vs Random      Result: 94 to 6
Match 2: Student vs MM_Null     Result: 93 to 7
Match 3: Student vs MM_Open     Result: 86 to 14
Match 4: Student vs MM_Improved Result: 81 to 19
Match 5: Student vs AB_Null     Result: 75 to 25
Match 6: Student vs AB_Open     Result: 62 to 38
Match 7: Student vs AB_Improved Result: 67 to 33
```

```
Results:
Student          79.71%
```

This function seems to be better function as expected.

(3) The third function I use $\exp()$ to both of the two variables because I think the exp curves are steeper thus should more greatly restrict the opponent's moves. However, I think $\exp()$ function is a time-consuming function, so I do not expect this function to be better than last function. Below is one of the results with NUM_MATCHES = 25

```
*****
Evaluating: ID_Improved
*****

Playing Matches:
Match 1: ID_Improved vs Random      Result: 87 to 13
Match 2: ID_Improved vs MM_Null     Result: 85 to 15
Match 3: ID_Improved vs MM_Open     Result: 82 to 18
Match 4: ID_Improved vs MM_Improved Result: 78 to 22
Match 5: ID_Improved vs AB_Null     Result: 75 to 25
Match 6: ID_Improved vs AB_Open     Result: 59 to 41
Match 7: ID_Improved vs AB_Improved Result: 50 to 50

Results:
ID_Improved      73.71%

*****
Evaluating: Student
*****

Playing Matches:
Match 1: Student vs Random      Result: 86 to 14
Match 2: Student vs MM_Null     Result: 88 to 12
Match 3: Student vs MM_Open     Result: 80 to 20
Match 4: Student vs MM_Improved Result: 72 to 28
Match 5: Student vs AB_Null     Result: 64 to 36
Match 6: Student vs AB_Open     Result: 46 to 54
Match 7: Student vs AB_Improved Result: 53 to 47

Results:
Student          69.86%
```

As above result shows, this function did not perform well, but this is not a surprise. The time consumed on heuristic function calculation is too much.

So finally I choose function 2 $\#my_move / \#oppo_move$ as my heuristic function. Because (1) it performs better than the other two. (2) the calculation is easy and straightforward and easy to implement. (3) the steep curve of $\#my_move / \#oppo_move$ will reduce the number of opponent's move faster than the linear curve of $\#my_move$ -

#oppo_move, but less overhead than the exponential functions. So I recommend using #my_move / #oppo_move as heuristic function.

Revised on 4/4/2017 as per suggestion from the reviewer:

Table 1 shows a summary of the comparison of the three heuristic functions. For each heuristic function, the tournament was executed five times (five rounds). We can see from the table that heuristic function (2) consistently performed better than ID_Improved, and it's the best heuristic function among the three.

Table 1. Heuristic functions comparison result

	Round	ID_Improved	Student
Heuristic Function (1)	1	71.43%	69.29%
	2	76.43%	72.14%
	3	70.71%	77.86%
	4	71.43%	68.57%
	5	74.29%	73.57%
	Average	72.86%	72.29%
Heuristic Function (2)	1	71.43%	80.00%
	2	72.86%	78.57%
	3	76.43%	78.57%
	4	68.57%	86.43%
	5	67.86%	81.43%
	Average	71.43%	81.00%
Heuristic Function (3)	1	74.29%	71.43%
	2	73.57%	67.14%
	3	72.14%	69.29%
	4	69.29%	72.14%
	5	75.71%	65.00%
	Average	73.00%	69.00%