Heuristic Function

Ren Silva
renato_s_silva@hotmail.com
+61 408 060504

I tested my agent in a Macbook Air 13" - so not a very powerful CPU. My searches were always running out of time and returning sub-optimal moves. I guess it is a great way to test an algorithm.

I ended up choosing: *Number of My Moves minus Number of Opponent's Moves.*

These were my attempts to select the heuristic function:

Number of My Moves

My first attempt of a heuristic function was to use *Number of my moves.* This was the function that was used in class for most examples, and a great starting point.

The result was that my agent was slightly worse than 'ID_Improved' agent, as shown below:

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

Playing Matches:
----------
  Match 1: ID_Improved vs   Random    Result: 16 to 4
  Match 2: ID_Improved vs   MM_Null   Result: 18 to 2
  Match 3: ID_Improved vs   MM_Open   Result: 12 to 8
  Match 4: ID_Improved vs MM_Improved Result: 13 to 7
  Match 5: ID_Improved vs   AB_Null   Result: 19 to 1
  Match 6: ID_Improved vs   AB_Open   Result: 9 to 11
  Match 7: ID_Improved vs AB_Improved Result: 11 to 9


Results:
----------
ID_Improved          70.00%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random    Result: 18 to 2
  Match 2:   Student   vs   MM_Null   Result: 18 to 2
  Match 3:   Student   vs   MM_Open   Result: 9 to 11
  Match 4:   Student   vs MM_Improved Result: 10 to 10
  Match 5:   Student   vs   AB_Null   Result: 17 to 3
  Match 6:   Student   vs   AB_Open   Result: 12 to 8
  Match 7:   Student   vs AB_Improved Result: 10 to 10
```

```
Results:
----------
Student                 67.14%
```

Number of My Moves minus Number of Opponent's Moves

I then changed it to "Number of my moves" minus "Number of Opponent's moves". This has led to significant improvement of my agent, as shown below:

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

Playing Matches:
----------
  Match 1: ID_Improved vs   Random    Result: 16 to 4
  Match 2: ID_Improved vs   MM_Null   Result: 17 to 3
  Match 3: ID_Improved vs   MM_Open   Result: 10 to 10
  Match 4: ID_Improved vs MM_Improved Result: 7 to 13
  Match 5: ID_Improved vs   AB_Null   Result: 19 to 1
  Match 6: ID_Improved vs   AB_Open   Result: 9 to 11
  Match 7: ID_Improved vs AB_Improved Result: 13 to 7


Results:
----------
ID_Improved             65.00%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random    Result: 18 to 2
  Match 2:   Student   vs   MM_Null   Result: 19 to 1
  Match 3:   Student   vs   MM_Open   Result: 15 to 5
tournament.py:100: UserWarning: One or more agents lost a match this
round due to timeout. The get_move() function must return before
time_left() reaches 0 ms. You will need to leave some time for the
function to return, and may need to increase this margin to avoid
timeouts during  tournament play.
  warnings.warn(TIMEOUT_WARNING)
  Match 4:   Student   vs MM_Improved Result: 14 to 6
  Match 5:   Student   vs   AB_Null   Result: 17 to 3
  Match 6:   Student   vs   AB_Open   Result: 16 to 4
  Match 7:   Student   vs AB_Improved Result: 13 to 7


Results:
----------
Student                 80.00%
```

Even though I have gained some significant improvement, my agent ran out of time in one of the matches, without returning a move. I went back to the code and found out why - and fixed it.

Number of My Moves minus 2 x Number of Opponent's Moves

I then went and weighted the opponent's moves more, like it was suggested in class.

The improvement was not significant, 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: 19 to 1
  Match 3: ID_Improved vs   MM_Open   Result: 11 to 9
  Match 4: ID_Improved vs MM_Improved Result: 13 to 7
  Match 5: ID_Improved vs   AB_Null   Result: 17 to 3
  Match 6: ID_Improved vs   AB_Open   Result: 8 to 12
  Match 7: ID_Improved vs AB_Improved Result: 13 to 7


Results:
----------
ID_Improved           70.71%

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

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


Results:
----------
Student               71.43%
```

This could have been just some random variation, so I tried again:

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

Playing Matches:
----------
  Match 1: ID_Improved vs   Random    Result: 18 to 2
```

```
  Match 2: ID_Improved vs   MM_Null   Result: 18 to 2
  Match 3: ID_Improved vs   MM_Open   Result: 13 to 7
  Match 4: ID_Improved vs MM_Improved Result: 11 to 9
  Match 5: ID_Improved vs   AB_Null   Result: 19 to 1
  Match 6: ID_Improved vs   AB_Open   Result: 13 to 7
  Match 7: ID_Improved vs AB_Improved Result: 13 to 7


Results:
----------
ID_Improved         75.00%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random    Result: 17 to 3
  Match 2:   Student   vs   MM_Null   Result: 20 to 0
  Match 3:   Student   vs   MM_Open   Result: 15 to 5
  Match 4:   Student   vs MM_Improved Result: 11 to 9
  Match 5:   Student   vs   AB_Null   Result: 18 to 2
  Match 6:   Student   vs   AB_Open   Result: 12 to 8
  Match 7:   Student   vs AB_Improved Result: 8 to 12


Results:
----------
Student             72.14%
```

So, this is definitely not the best heuristic function.

(Back to) Number of My Moves minus Number of Opponent's Moves

So, I went back to option 2, as it appears to be better.

Indeed, my agent's best game occurs when it is focused in winning at the same time that it tries to stop the opponent from wining.

```
**************************
 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: 11 to 9
  Match 4: ID_Improved vs MM_Improved Result: 10 to 10
  Match 5: ID_Improved vs   AB_Null   Result: 20 to 0
  Match 6: ID_Improved vs   AB_Open   Result: 14 to 6
  Match 7: ID_Improved vs AB_Improved Result: 15 to 5
```

```
Results:
----------
ID_Improved           75.00%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random    Result: 19 to 1
  Match 2:   Student   vs   MM_Null   Result: 19 to 1
  Match 3:   Student   vs   MM_Open   Result: 16 to 4
  Match 4:   Student   vs MM_Improved Result: 13 to 7
  Match 5:   Student   vs   AB_Null   Result: 16 to 4
  Match 6:   Student   vs   AB_Open   Result: 17 to 3
  Match 7:   Student   vs AB_Improved Result: 11 to 9


Results:
----------
Student               79.29%
```

2 x Number of My Moves minus Number of Opponent's Moves

And, in, one last attempt, I tried to weigh "my moves" (multiplying to by 2).

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

Playing Matches:
----------
  Match 1: ID_Improved vs   Random    Result: 17 to 3
  Match 2: ID_Improved vs   MM_Null   Result: 18 to 2
  Match 3: ID_Improved vs   MM_Open   Result: 10 to 10
  Match 4: ID_Improved vs MM_Improved Result: 14 to 6
  Match 5: ID_Improved vs   AB_Null   Result: 15 to 5
  Match 6: ID_Improved vs   AB_Open   Result: 12 to 8
  Match 7: ID_Improved vs AB_Improved Result: 14 to 6


Results:
----------
ID_Improved           71.43%

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

Playing Matches:
----------
tournament.py:100: UserWarning: One or more agents lost a match this
round due to timeout. The get_move() function must return before
time_left() reaches 0 ms. You will need to leave some time for the
```

```
function to return, and may need to increase this margin to avoid
timeouts during  tournament play.
  warnings.warn(TIMEOUT_WARNING)
  Match 1:   Student   vs   Random    Result: 18 to 2
  Match 2:   Student   vs   MM_Null   Result: 16 to 4
  Match 3:   Student   vs   MM_Open   Result: 11 to 9
  Match 4:   Student   vs MM_Improved Result: 14 to 6
  Match 5:   Student   vs   AB_Null   Result: 18 to 2
  Match 6:   Student   vs   AB_Open   Result: 10 to 10
  Match 7:   Student   vs AB_Improved Result: 10 to 10


Results:
----------
Student             69.29%
```

The result was disappointing. I tried once more:

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

Playing Matches:
----------
  Match 1: ID_Improved vs   Random    Result: 19 to 1
  Match 2: ID_Improved vs   MM_Null   Result: 18 to 2
  Match 3: ID_Improved vs   MM_Open   Result: 11 to 9
  Match 4: ID_Improved vs MM_Improved Result: 12 to 8
  Match 5: ID_Improved vs   AB_Null   Result: 19 to 1
  Match 6: ID_Improved vs   AB_Open   Result: 10 to 10
  Match 7: ID_Improved vs AB_Improved Result: 12 to 8


Results:
----------
ID_Improved         72.14%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random    Result: 18 to 2
  Match 2:   Student   vs   MM_Null   Result: 16 to 4
  Match 3:   Student   vs   MM_Open   Result: 12 to 8
  Match 4:   Student   vs MM_Improved Result: 12 to 8
  Match 5:   Student   vs   AB_Null   Result: 18 to 2
  Match 6:   Student   vs   AB_Open   Result: 13 to 7
  Match 7:   Student   vs AB_Improved Result: 11 to 9


Results:
----------
Student             71.43%
```

The result was still disappointing. I went back to option 2:

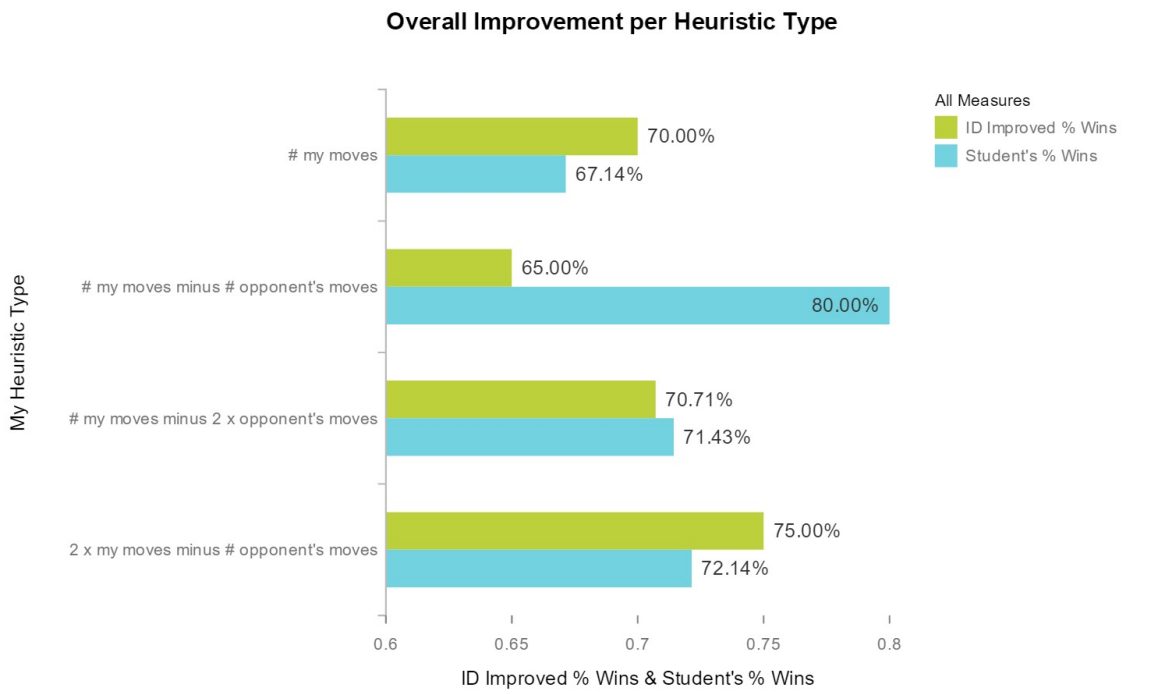
(Back to) Number of My Moves minus Number of Opponent's Moves

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

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


Results:
----------
ID_Improved           67.86%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random    Result: 17 to 3
  Match 2:   Student   vs   MM_Null   Result: 18 to 2
  Match 3:   Student   vs   MM_Open   Result: 16 to 4
  Match 4:   Student   vs MM_Improved Result: 13 to 7
  Match 5:   Student   vs   AB_Null   Result: 19 to 1
  Match 6:   Student   vs   AB_Open   Result: 16 to 4
  Match 7:   Student   vs AB_Improved Result: 10 to 10


Results:
----------
Student               77.86%
(aind) Rens-MacBook-Air:AIND-Isolation Ren$
```

The results are summarized on the visualization below:

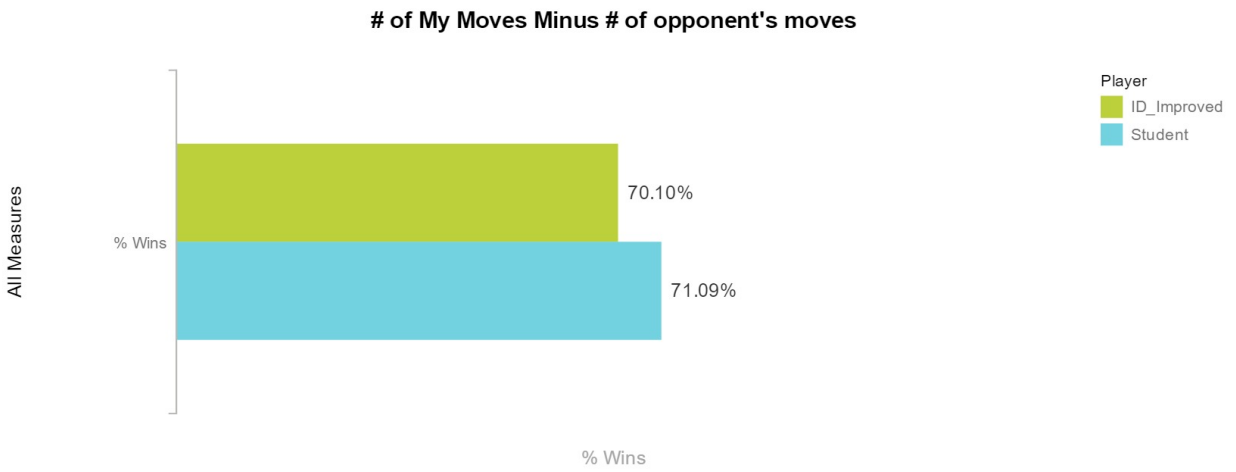
Choosing a Heuristic Function

The best improvement was achieved by using the function 'Number of My Moves Minus Number of Opponent's Moves' - the Student player won 80% of the times when compared with the benchmark 65%



Why I selected *My moves minus Opponent's Moves*

I then ran the Tournament script 38 times to confirm that it would consistently produce good results. While it did not produce the same results as originally (bringing the number of wins from 65% to 80%), there was consistent improvement over the benchmark "ID Improved" measure, as shown in the visualization below:



Running "Number of My Moves Minus Number of Opponent's Moves" over 38 tournaments produced positive results (the student was consistently better), but not was encouraging as the 80% over 65% that I obtained the first time.

Looking at the game it is not difficult to understand why this heuristic function is the best one:

1. The heuristic accurately balances the need to be aggressive (i.e. choosing winning moves) with being defensive (choosing moves that make it harder for the opponent to pick a winning move). Any other choice would have favoured one option over the other.

2. It is a simple calculation - so, the agent will have the opportunity to calculate this simple function many times before running out of time over an iterative deepening cycle. This function is less simple than the first option (number of my moves), but is simpler than the others that involve multiplications. This may appear irrelevant, but when the agent may have to calculate this function millions (or billions) of times, a simple function allows for more trials within the available time.

3. And obviously - this heuristic function has proven itself. It has consistently produced better results than the others. I also tried the function on a faster computer - my MacBook Pro with faster CPU - and the results were consistent - the level of improvement was the same. As my former manager used to say, there is nothing like "runs on the board" to prove a point.