AIND-Isolation Heuristic Analysis

March 18, 2017

1 Results Comparision

```
In [2]: import tournament as tn
```

```
In [3]: def custom_score(game, player):
            if game.is_loser(player):
                return float("-inf")
            if game.is_winner(player):
                return float("inf")
            my_moves = game.get_legal_moves(player)
            oppo_moves = game.get_legal_moves(game.get_opponent(player))
            return float(len(my_moves)-len(oppo_moves))
        tn.brief(custom_score)
```

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

Playing Matches:
----------
  Match 1:   Student   vs   Random            Result: 20 to 0
  Match 2:   Student   vs   MM_Null           Result: 20 to 0
  Match 3:   Student   vs   MM_Open           Result: 20 to 0
  Match 4:   Student   vs MM_Improved         Result: 20 to 0
  Match 5:   Student   vs   AB_Null           Result: 20 to 0
  Match 6:   Student   vs   AB_Open           Result: 19 to 1
  Match 7:   Student   vs AB_Improved         Result: 20 to 0


Results:
----------
Student             99.29%
```

The evaluation function used above:

$$f = m - o$$

, where m is the number of legal move of the player, s is the number of legal move of the oppoenent.

```
In [4]: def custom_score2(game, player):
            if game.is_loser(player):
                return float("-inf")
            if game.is_winner(player):
                return float("inf")
            my_moves = game.get_legal_moves(player)
            oppo_moves = game.get_legal_moves(game.get_opponent(player))
            return float(len(my_moves)**2-len(oppo_moves)**2)
        tn.brief(custom_score2)


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

Playing Matches:
----------
  Match 1:   Student   vs   Random            Result: 20 to 0
  Match 2:   Student   vs   MM_Null           Result: 20 to 0
  Match 3:   Student   vs   MM_Open           Result: 20 to 0
  Match 4:   Student   vs MM_Improved         Result: 20 to 0
  Match 5:   Student   vs   AB_Null           Result: 20 to 0
  Match 6:   Student   vs   AB_Open           Result: 20 to 0
  Match 7:   Student   vs AB_Improved         Result: 20 to 0


Results:
----------
Student             100.00%
```

The Evaluation function used above:

$$f = m^2 - o^2$$

, where m is the number of legal move of the player, s is the number of legal move of the oppoenent.

```
In [5]: def custom_score3(game, player):
            if game.is_loser(player):
                return float("-inf")
            if game.is_winner(player):
                return float("inf")
            my_moves = game.get_legal_moves(player)
            oppo_moves = game.get_legal_moves(game.get_opponent(player))
            return float(len(my_moves)-2*len(oppo_moves))
        tn.brief(custom_score3)
```

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

Playing Matches:
----------
  Match 1:   Student   vs   Random            Result: 20 to 0
  Match 2:   Student   vs   MM_Null           Result: 20 to 0
  Match 3:   Student   vs   MM_Open           Result: 20 to 0
  Match 4:   Student   vs MM_Improved         Result: 19 to 1
  Match 5:   Student   vs   AB_Null           Result: 19 to 1
  Match 6:   Student   vs   AB_Open           Result: 20 to 0
  Match 7:   Student   vs AB_Improved         Result: 20 to 0


Results:
----------
Student             98.57%
```

The evaluation function used above:

$$f = m - 2o$$

, where m is the number of legal move of the player, s is the number of legal move of the oppoenent.

2 Summary

- The first evaluation function is a linear function that rewards player's legal moves and equally penalize the oppoents' legal move.
- The second evaluation function is a quadratic function that rewards player's legal moves and penalize the oppoents' legal move.
- The third evaluation function is a linear function that rewards player's legal moves and aggressively penalize the oppoents' legal move.

Based on the running result, the second evaluation function $f = m^2 - o^2$ seems to be the best. Since the result are actually very close, more iternations of matches should be carried out to draw any conclusion.