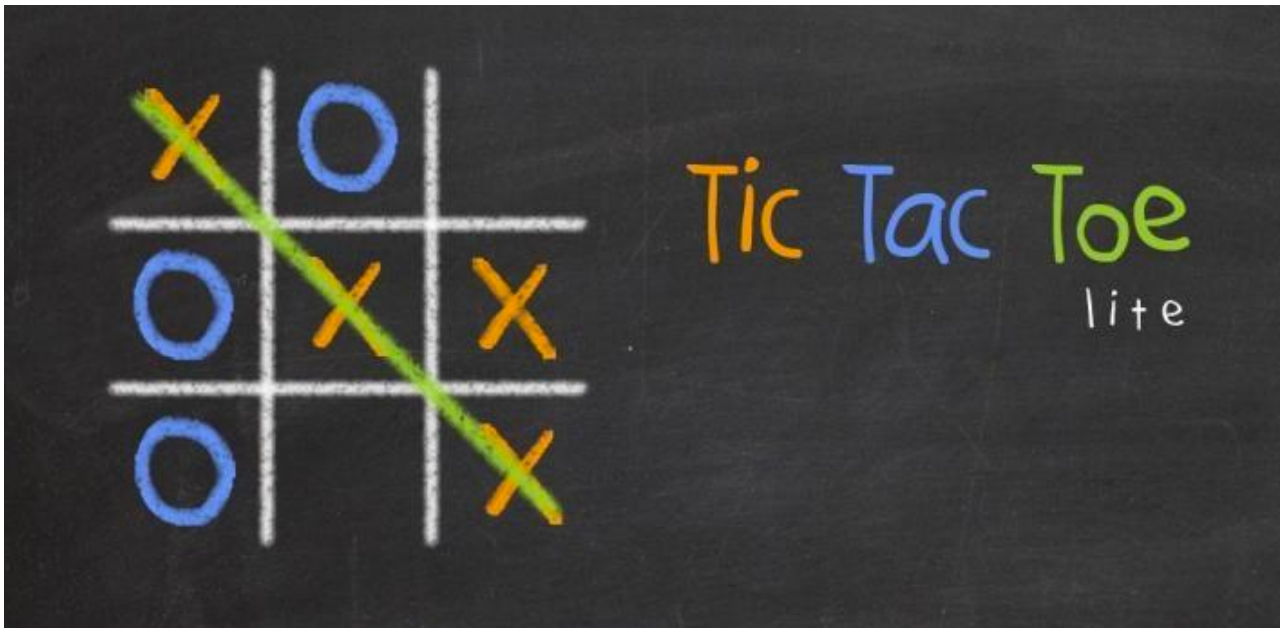
Build a Game-Playing Agent

Heuristic Analysis

Dave Fang / dave.fang@outlook.com



Build a Game-Playing Agent

In this project, we are required to build a game-playing agent of Isolation, using two kind of strategies which is Minimax and Alpha-Beta. And we should evaluate the scoring functions we invented.

Heuristic I

Source Code:

```
def custom_score(game, player):

      if game.is_loser(player):

          return float("-inf")

      if game.is_winner(player):

          return float("inf")

      own_moves = len(game.get_legal_moves(player))

      opp_moves = len(game.get_legal_moves(game.get_opponent(player)))

      return float(own_moves - opp_moves)
```

This heuristic function uses the difference between legal moves of opponent and legal moves of current player. It is a good scoring function, which we should make every effort to enlarge the difference.

Heuristic II

Source Code:

```
def custom_score_2(game, player):

    if game.is_loser(player):

        return float("-inf")

    if game.is_winner(player):

        return float("inf")

    return float(len(game.get_legal_moves(player)))
```

This heuristic function is about the number of legal moves of current player. We should try our best to maximize the number of legal moves.

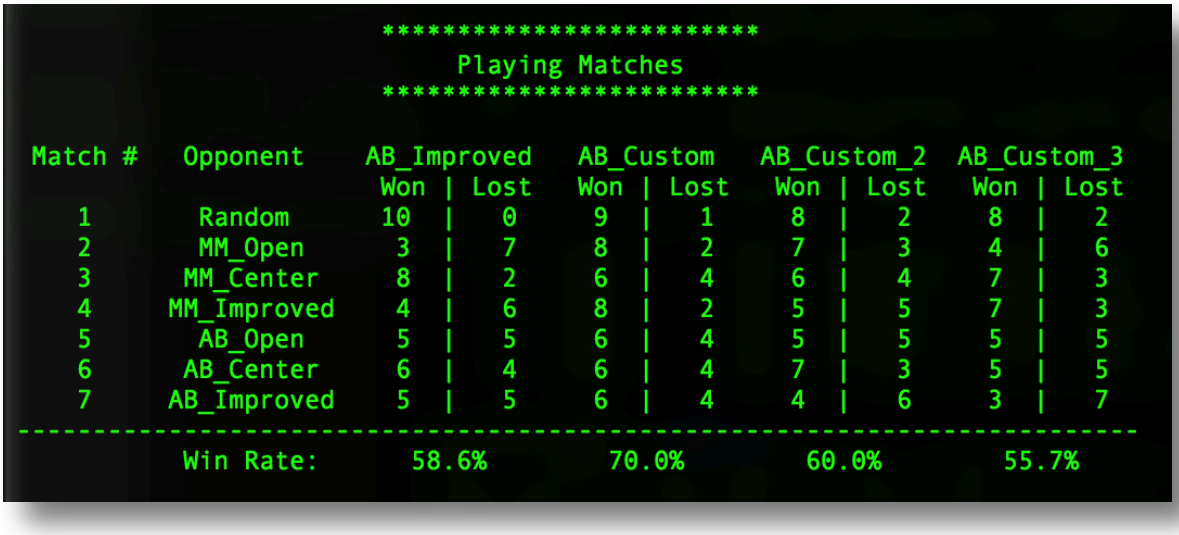
Heuristic III

Source Code:

```
def custom_score_3(game, player):

    if game.is_loser(player):

        return float("-inf")

    if game.is_winner(player):

        return float("inf")

    w, h = game.width / 2., game.height / 2.

    y, x = game.get_player_location(player)

    return float((h - y) ** 2 + (w - x) ** 2)
```

This heuristic function is kind of wild guess; it is not a solid scoring function because it is only about the location of current player.

Performance



```
**************************
    Playing Matches
**************************

Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                       Won | Lost   Won | Lost   Won | Lost   Won | Lost
   1       Random      10  |   0     9  |   1     8  |   2     8  |   2
   2       MM_Open      3  |   7     8  |   2     7  |   3     4  |   6
   3      MM_Center     8  |   2     6  |   4     6  |   4     7  |   3
   4     MM_Improved    4  |   6     8  |   2     5  |   5     7  |   3
   5       AB_Open      5  |   5     6  |   4     5  |   5     5  |   5
   6      AB_Center     6  |   4     6  |   4     7  |   3     5  |   5
   7     AB_Improved    5  |   5     6  |   4     4  |   6     3  |   7
--------------------------------------------------------------------------
          Win Rate:      58.6%        70.0%        60.0%        55.7%
```

For all three heuristic functions, the first one is the best which has 70% winning rate. The reason Heuristic I can get best performance is that the calculation is easy, only do the subtraction, which can reduce consuming time and make it go faster and deeper. So I do recommend first heuristic function. On the one hand, it is easy to understand that to enlarge current player's number of legal option and minimize the opponent. On the other hand, it is a tradeoff between performance and performance overhead, which is easy to calculate and go deeper.