Artificial Intelligence Nanodegree Program

Project II Build an Adversarial Search Agent

Heuristic Analysis

Luis Palacios

Three different heuristics to calculate the value of a game state from the point of view of the given player were implemented in order to evaluate the performance of each of them against the heuristic thought in class which consist on calculating the difference between the legal moves of player v.s. the opponent. This heuristic is called “ID_Improved” where “ID” means iterative deepening. At least one heuristic must outperforms ID_Improved. This paper describes the results achieved by using the `tournament.py` script to evaluate the effectiveness of the custom heuristics.

Heuristic 1 “Dangerous Chase”
This heuristic favor the players moves which are closer to the opponent this is accomplished through getting the min distance between the player and the opponent it returns the negative of the absolute difference between the sum of the locations, smaller distance means higher scores as a result player moves which have the least distance from the opponent location will have a higher score

Heuristic 2 “Run boy run”
This heuristics elaborates on ID_Improved, but this time it favors player moves which are farther away from its opponent. This is done by adding the Euclidian distance from player to opponent to the number of legal moves of the player. As a result, player moves which have a greater distance from the opponent’s location will have a higher score and thus be favored by this heuristic.

Heuristic 3 “Look ahead”
This is the best heuristic implemented it also elaborates on the ID_Improved. The main difference between the ID_Improved and the Look Ahead heuristic is that in addition to calculating the difference between legal moves of player v.s. the opponent, it forecast each player move and calculates how many moves does each of those next legal moves have this this heuristic favors legal moves which have a larger number of moves in the future.

After running multiple times the script tournament.py the results were always very close to what the following pictures describes:

```
(aind) C:\Users\Luis Palacios\source\repos\isolation-test>python tournament.py

This script evaluates the performance of the custom_score evaluation
function against a baseline agent using alpha-beta search and iterative
deepening (ID) called `AB_Improved`. The three `AB_Custom` agents use
ID and alpha-beta search with the custom_score functions defined in
game_agent.py.

                    *************************
                         Playing Matches
                    *************************

 Match #   Opponent    AB_Improved   LookAhead    RunBoyRun  DangerousChase
                        Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1       Random       9  |   1     9  |   1    10  |   0     9  |   1
    2       MM_Open      6  |   4     8  |   2     8  |   2     7  |   3
    3      MM_Center     9  |   1    10  |   0    10  |   0     9  |   1
    4     MM_Improved    9  |   1     9  |   1     6  |   4     6  |   4
    5       AB_Open      4  |   6     5  |   5     5  |   5     4  |   6
    6      AB_Center     8  |   2     7  |   3     6  |   4     5  |   5
    7     AB_Improved    7  |   3     8  |   2     5  |   5     6  |   4
--------------------------------------------------------------------------
          Win Rate:      74.3%        80.0%        71.4%        65.7%
```

```
(aind) C:\Users\Luis Palacios\source\repos\isolation-test>python tournament.py

This script evaluates the performance of the custom_score evaluation
function against a baseline agent using alpha-beta search and iterative
deepening (ID) called `AB_Improved`. The three `AB_Custom` agents use
ID and alpha-beta search with the custom_score functions defined in
game_agent.py.

                    *************************
                         Playing Matches
                    *************************

 Match #   Opponent    AB_Improved   LookAhead    RunBoyRun  DangerousChase
                        Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1       Random       8  |   2     9  |   1    10  |   0     9  |   1
    2       MM_Open      9  |   1     9  |   1     9  |   1     9  |   1
    3      MM_Center     9  |   1    10  |   0     8  |   2     9  |   1
    4     MM_Improved    6  |   4     7  |   3     7  |   3     5  |   5
    5       AB_Open      3  |   7     6  |   4     5  |   5     5  |   5
    6      AB_Center     9  |   1     4  |   6     5  |   5     5  |   5
    7     AB_Improved    3  |   7     5  |   5     6  |   4     5  |   5
--------------------------------------------------------------------------
          Win Rate:      67.1%        71.4%        71.4%        67.1%
```

The Look Ahead heuristic was selected as the best heuristic implemented since it was the one that consistently beat the winning rate of ID_Improved and even ID_Improved itself. Some of the reasons why the Look Ahead heuristics performs consistently better than others are:

1. Elaborates on ID_Improved, which is an already good and simple heuristic
2. It continue to be a simple heuristic, and it does not affect the ability of the algorithm to search deep in the tree
3. It takes into consideration what the set of legal moves holds available for the next phase of the game play