Heuristic Analysis

Objective

The objective of this work is to generate, implement, analyze, and compare three custom heuristic functions used to calculate the value of an *Isolation* board. These functions are compared against the *Improved* heuristic while being executed by an agent using the *Iterative deepening* search algorithm. This agent is called the **ID_Improved** agent.

Both agents are excuted in a tournament against other three agents each of them using a different search algorithm and with three different heuristic functions:

- **Random agent**: It is an agent that choses randomly. This agent is not executed with any herustic function (as it does not uses any).
- **Minimax Agent (MM)**: It is an agent that implements a fixed-depth minimax search algorithm.
- **Alpha-Beta agent (AB)**: It is an agent that implements the fixed-depth Alpha-Beta pruning search algorithm.

The **AB** and **MM** agents are evaluated with three different heuristics:

- **Null heuristic**: This heuristic presumes no knowledge for non-terminal states, and returns the same uninformative value for all other states.
- **Open move heuristic**: This heuristic implements the basic evaluation function described in lecture that outputs a score equal to the number of available moves for the player.
- **Improved heuristic**: This heuristic calculates the difference of available moves between the two players.

The tournament estimates the strength rating of the student-agent with iterative deepening and a custom heuristic evaluation function against the **AB** and **MM** test agents by running a round-robin tournament.

The *student agent* and the *ID_Improved* agents play 20 "fair" matches against each test agent. The matches are fair because, for each match, the board is initialized randomly for both players, and the players play each match twice -- switching the player order between games. This helps to correct for imbalances in the game due to both starting position and initiative.

The results for the *ID_Improved* agent were pretty stable on the computer used to exeuted the tournaments. An example of a tipical result is the following:

```
Match 1: ID_Improved vs   Random      Result: 40 to 0
Match 2: ID_Improved vs   MM_Null     Result: 38 to 2
Match 3: ID_Improved vs   MM_Open     Result: 30 to 10
```

```
Match 4: ID_Improved vs MM_Improved   Result: 25 to 15
Match 5: ID_Improved vs   AB_Null     Result: 31 to 9
Match 6: ID_Improved vs   AB_Open     Result: 28 to 12
Match 7: ID_Improved vs AB_Improved   Result: 26 to 14
ID_Improved          77.86%
```

This agent played in total 4200 games and won 3257 of them giving it a total score of 77.54%

Heuristics Analyzed

Available moves in common

This heuristic function calculates the size of the intersection of the sets of open moves of each player. The board score is positive if the active player is the current player and negative otherwise.

The rationale behind this heuristic is that the moves in common are the moves were the active agent can close options for the opponent possibly creating a partition. Thus, when the active player is our agent the score is positive meaning that from this board some good moves can be possible done and viceversa.

A tipical tournament for this agent was as follows:

```
Match 1:    Student    vs    Random      Result: 30 to 10
Match 2:    Student    vs    MM_Null     Result: 24 to 16
Match 3:    Student    vs    MM_Open     Result: 24 to 16
Match 4:    Student    vs MM_Improved    Result: 15 to 25
Match 5:    Student    vs    AB_Null     Result: 25 to 15
Match 6:    Student    vs    AB_Open     Result: 21 to 19
Match 7:    Student    vs AB_Improved    Result: 17 to 23
Student              55.71%
```

This agent played in total 1400 games and won 777 of them giving it a total score of 55.50% which is extremelly lower than the *ID_Improved* agent's score.

Distance to the opponent

This heuristic function calculates the distance between the two players.

The rationale behind this heuristic is that it could be better to be further away from the opponent and prevent the blocking of movements.

A tipical tournament for this agent was as follows:

```
Match 1:    Student    vs    Random      Result: 32 to 8
Match 2:    Student    vs    MM_Null     Result: 26 to 14
Match 3:    Student    vs    MM_Open     Result: 17 to 23
Match 4:    Student    vs MM_Improved    Result: 19 to 21
Match 5:    Student    vs    AB_Null     Result: 24 to 16
```

```
Match 6:    Student    vs    AB_Open     Result: 22 to 18
Match 7:    Student    vs AB_Improved    Result: 18 to 22
Student               56.43%
```

This agent played in total 1400 games and won 789 of them giving it a total score of 56.35% which is better than the *common moves* agent, but still a lot worse than the *ID_Improved* agent.

Linear combination of moves

The heuristics presented in the lectures are based on the number of available moves that each agent has with a preference for the boards that have more available movements for the player. It is suggested that other combinations can be implemented assigning different weights to each player moves creating a family of heuristics generated by the linear combination of the variables:

```
x - amount of open moves of the player
y - amount of open moves of the opponent
```

Each heuristic has the form:

```
h = a*x + b*y + c
```

It is possible to perform a grid search over the values of a,b,c to find the values that maximize the heuristic score.

A grid search was performed restricting the possible values of a,b, and c to following ranges:

```
-3 <= x,y <= 3
c in {-24, -16, -8, 0, 8, 16, 24}
```

The selected ranges are mostly arbitrary and were kept small in order to be able to execute the search in reasonable time.

The evaluation of each heuristic was performed by running a tournament of 100 matches against the *AB* agent and counting the winning matches. The results of the whole grid search can be found in the file grid_search_results.txt (https://github.com/edhzsz/AIND-Isolation/blob/master/grid_search_results.txt) in this repository.

The best values found are: a = 1 b = -2 c = -8 Which got a score of 75.33%.

It is worth noting that the *open moves* and *improved* heuristics described in the lectures were also evaluated and they got a score of 67.66% and 68.33% respectively.

An agent was generated and tested using this values and the results of a tipical tournament for this agent are as follows:

```
Match 1:    Student    vs    Random      Result: 37 to 3
Match 2:    Student    vs    MM_Null     Result: 37 to 3
Match 3:    Student    vs    MM_Open     Result: 31 to 9
Match 4:    Student    vs MM_Improved    Result: 29 to 11
```

```
Match 5:    Student    vs    AB_Null      Result: 35 to 5
Match 6:    Student    vs    AB_Open      Result: 27 to 13
Match 7:    Student    vs AB_Improved    Result: 29 to 11
Student              80.36%
```

This agent played in total 1400 games and won 1119 of them giving it a total score of 79.92% which is better than the *ID_Improved* agent's score making it suitable to be used as the evaluation function by an agent that plays the Isolation game with horses.

Conclusion

The results of all the agents can be summarized on the following table:

Agent	Games played	Games Won	Games Lost	Score
ID_Improved	4200	3257	943	77.54%
Available moves In Common	1400	777	623	55.50%
Distance to the opponent	1400	789	611	56.35%
Linear combination of moves	1400	1119	181	79.92%

I recommend to use the ***Linear combination of moves*** agent because:

- It is the only proposed heuristic that was able to outperform the ***ID_Improved*** agent.
- Being a linear combination of open moves of both the player and the opponent, it is as complex as the ***Improved*** heuristic and it is not slower to calculate.
- It is easy to see that it considers both players to calculate the score giving a stronger (negative) weight to the open moves of the opponent. Although it is not easy to describe exactly what is being messured.
- Both, ***ID_Improved*** and the ***Linear combination*** agents, traverse the tree to basically the same depth. The maximum depth for both is 19 and the minimum is 6. The average depth being traversed by the ***Id_Improved*** is 8,08 and it is 8,43 for the ***Linear combination*** agent. This supports the claim that the complexity of both heuristics is similar, but also means that the ***Linear combination*** heuristic estimates better the value of the board and predicts better the final outcome.