Heuristic Analysis

As part of the Project 2 of the AIND " Build a Game-Playing Agent" I had to implement the minimax, alpha-beta pruning and iterative deepening for an Isolation game. Also, in order to increase its win ratio I implemented some evaluations functions, which I will explain in detail:

**Heuristics**

1. Heuristic 1:

   Following the same intuition presented in the lecture, I decided to try an evaluation function that uses my moves available and my opponent moves available:

   $$score = my\_moves - a * opp\_moves$$

   *my_moves*: Legal moves from my position
   *opp_moves*: Legal moves from my opponent's position

```
def heuristic  1(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 - 1.4*opp  moves)
```

Fig.1 Code used for heuristic 1.

After trying several combinations for "*a*", I found that using the value of 1.4 perform quite well. After that, I decided to test several times (Number of matches: 5) the evaluation function with 1.4 in order to be sure that it could make Student Agent win.

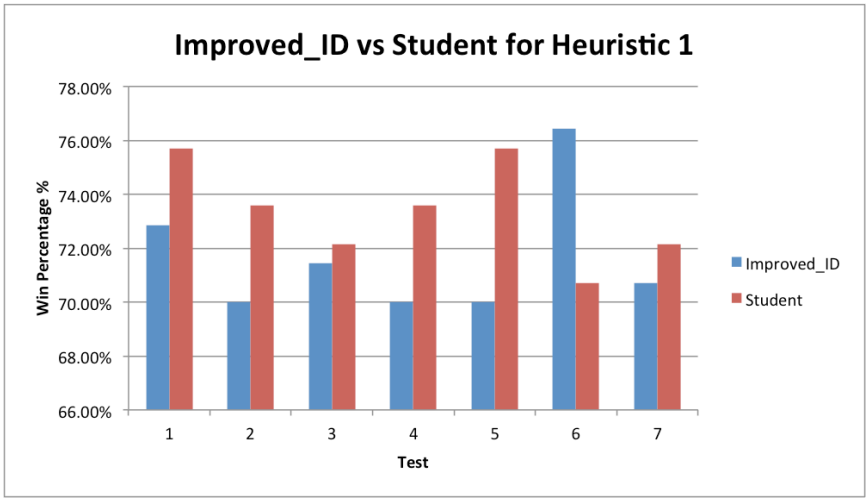


Fig.2 Chart comparing the performance of the coefficient 1.4.

As we can see in Fig.2, using 1.4 as the coefficient "a" will result in a win most of the time, having a maximum score of 75.71% (Fig.3). Notice that the number of matches was 20 (the variable is NUM_MATCHES is 5, but because of the rules it gets multiplied by 4)

The results of one of the test (number 5):

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

Playing Matches:
----------
  Match 1: ID_Improved vs   Random        Result: 17 to 3
  Match 2: ID  Improved vs   MM  Null      Result: 14 to 6
  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: 13 to 7
  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        70.00%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random        Result: 19 to 1
  Match 2:   Student   vs   MM_Null       Result: 15 to 5
  Match 3:   Student   vs   MM_Open       Result: 15 to 5
  Match 4:   Student   vs MM_Improved     Result: 13 to 7
  Match 5:   Student   vs   AB_Null       Result: 15 to 5
  Match 6:   Student   vs   AB_Open       Result: 12 to 8
  Match 7:   Student   vs AB_Improved     Result: 17 to 3


Results:
----------
Student           75.71%
```

Fig. 3: Results of one test using 1.4 coefficient.

2. Heuristic 2:

Looking the board, I realized that the center has more moves available than the edges, so I made a representation in order to know the number of moves for each position (Fig.4). As we can see the center has more moves available and the corners have the least.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 3 | 4 | 4 | 4 | 3 | 2 |
| 1 | 3 | 4 | 6 | 6 | 6 | 4 | 3 |
| 2 | 4 | 6 | 8 | 8 | 8 | 6 | 4 |
| 3 | 4 | 6 | 8 | 8 | 8 | 6 | 4 |
| 4 | 4 | 6 | 8 | 8 | 8 | 6 | 4 |
| 5 | 3 | 4 | 6 | 6 | 6 | 4 | 3 |
| 6 | 2 | 3 | 4 | 4 | 4 | 3 | 2 |

Figure 4: 7x7 board showing the number of legal moves available from a position.

This insight allowed me to develop a new heuristic; I listed all the positions that have 8, 6, 2 moves and I decided to consider the 3 and 4 moves as one category in order to develop a simple model. I was aware that this model was taking in account only the legal moves without considering the spaces occupied. I considered the model as follows:

$$score = a * (my\_moves - 1.4 * opp\_moves)$$

Where:

a: 1.5; if the position has 8 legal moves available.
1.3; if the position has 6 legal moves available.
1.2; if the position has 3 or 4 moves available.
1.0; if the position has 2 moves available.

Notice that I took the value of 1.4 (the coefficient of the opp_moves), since it allowed to me win the ID_Improved agent repeatedly before (Heuristic 1). The intuition behind this heuristic is to force the agent to move to the center or penalize if it is near to the corners; I tried several combinations but the one above was the only one that could win ID_Improved on several matches.

```
def heuristic_2(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)))

    list_center = [(2,2),(2,3),(2,4),(3,2),(3,3),(3,4),(4,2),(4,3),(4,4)]
    list_middle = [(2,1),(3,1),(4,1),(1,2),(1,3),(1,4),(2,5),(3,5),(4,5),(5,2),(5,3),(5,4)]
    list_outside = [(0,0),(0,6),(6,0),(6,6)]

    a=1
    if game.get_player_location(player) in list_center:
        a=1.5
    elif game.get_player_location(player) in list_middle:
        a=1.3
    elif game.get_player_location(player) in list_outside:
        a=1
    else:
        a=1.2

    return float(a*(own_moves - 1.4*opp_moves))
```

Fig. 5: Code for heuristic 2

I ran this model and I got the following result:

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

Playing Matches:
----------
  Match 1: ID  Improved vs   Random              Result: 20 to 0
  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: 8 to 12
  Match 5: ID_Improved vs   AB_Null             Result: 20 to 0
  Match 6: ID_Improved vs   AB_Open             Result: 12 to 8
  Match 7: ID_Improved vs AB_Improved           Result: 12 to 8


Results:
----------
ID_Improved        72.86%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random        Result: 20 to 0
  Match 2:   Student   vs   MM_Null       Result: 17 to 3
  Match 3:   Student   vs   MM  Open      Result: 13 to 7
  Match 4:   Student   vs MM_Improved             Result: 15 to 5
  Match 5:   Student   vs   AB  Null      Result: 16 to 4
  Match 6:   Student   vs   AB_Open       Result: 12 to 8
  Match 7:   Student   vs AB_Improved  Result: 12 to 8


Results:
----------
Student           75.00%
```

Fig. 6: Results form heuristic 2

3. Heuristic 3:

I tried another insight using the heuristic 1 but changing the "a" coefficient during the game. The intuition behind this was that during the first stage of the game using an evaluation such as my moves minus my opponent moves is not so much effective since both players have most of their legal moves available. However, as more moves are played the situation changes; the legal moves available are less than before so my approach was start chasing the opponent agent from that moment. After some moves of both players, our board (7x7) started to fill in, that is why I reduced the coefficient, to change the intensity of my agent. We can see this explanation in Fig. 7.

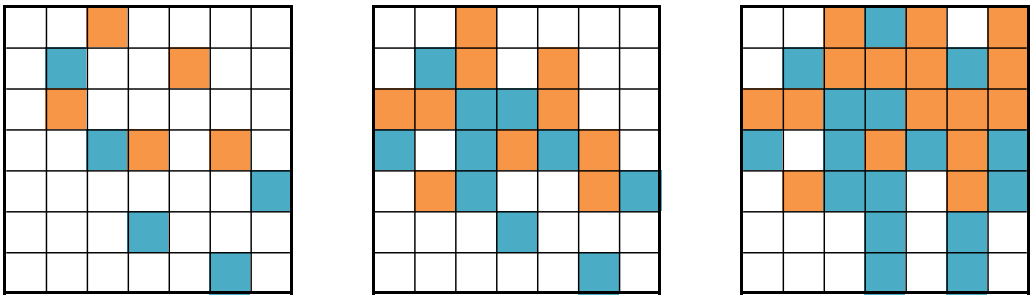


Figure 7: 7x7 board showing the states in the model proposed.

However, after some initial trials I realized that I have to define the thresholds for three states; in my case I chose the limit states when there are 10 and 29 occupied spaces, as we can see in the following figure (Fig. 8).

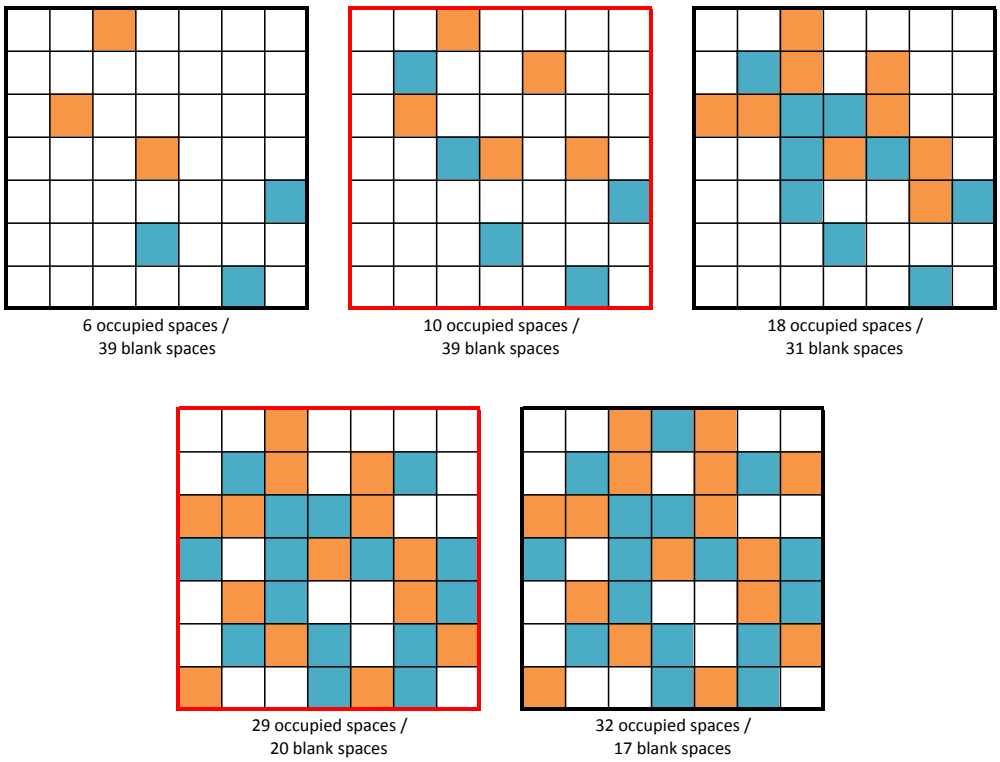


Figure 8: 7x7 board, the red borders show the moment when the coefficient will change.

Once I have indicated the thresholds, I have to give assign a value for the "a" value. Notice that I am using the formula below but "a" will change during the game.

$$score = my\_moves - a * opp\_moves$$

I made a table with all the combinations I tried, changing only the value for the middle-game which regulates how much my agent will chase ID_Improved.

| Conditions | | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|---|
| Coefficient "a" | >=39 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | >=20 | 2 | 2.2 | 2.4 | 2.5 | 2.6 | 2.7 | 2.8 | 3 |
| | <20 | 1.4 | 1.4 | 1.4 | 1.4 | 1.4 | 1.4 | 1.4 | 1.4 |

Table 1: Combinations for Heuristic 3.

After the tests, I got the following results:

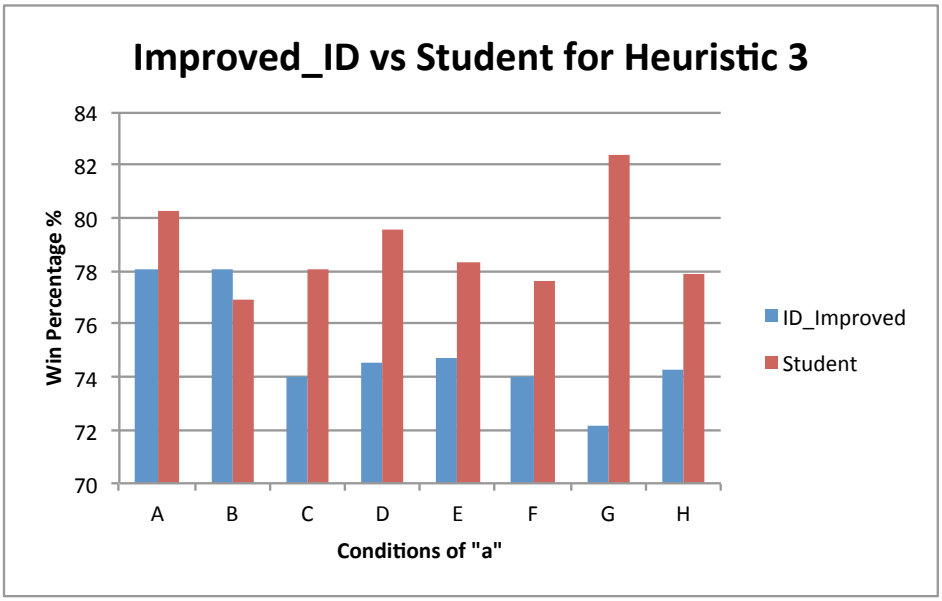


Fig. 9: Chart of the tests for Heuristic 3.

As we can see in the Fig. 9, the combinations "G", "A" and "D" performed better than the rest and ID_Improved. In the following table, we can see the details. Notice that for these tests I ran 60 simulations (NUM_MATCHES: 15)

| Coefficient | | | ID_Improved | Student | Name |
|---|---|---|---|---|---|
| A | >=39 | 1 | 78.10% | 80.24% | Heuristic 3a |
| | >=20 | 2 | | | |
| | <20 | 1.4 | | | |
| D | >=39 | 1 | 74.52% | 79.52% | Heuristic 3b |
| | >=20 | 2.5 | | | |
| | <20 | 1.4 | | | |
| G | >=39 | 1 | 72.14% | 82.38% | Heuristic 3c |
| | >=20 | 2.8 | | | |
| | <20 | 1.4 | | | |

Table 2: Summary of the best combinations using this model.

```
def heuristic_3c(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)))

    blank_spaces = len(game.get_blank_spaces())

    if blank_spaces >= 39:
        a=1
    elif blank_spaces >=20:
        a=2.8
    else:
        a=1.4

    return float(own  moves - a*opp  moves)
```

Fig. 10: Code for Heuristic 3c

I will compare these three heuristics with the ones above in order to determine what is the one that perform better.

## Comparisons

I ran 400 simulations (NUM_MATCHES: 100) for each heuristic specified below in Fig. 11.

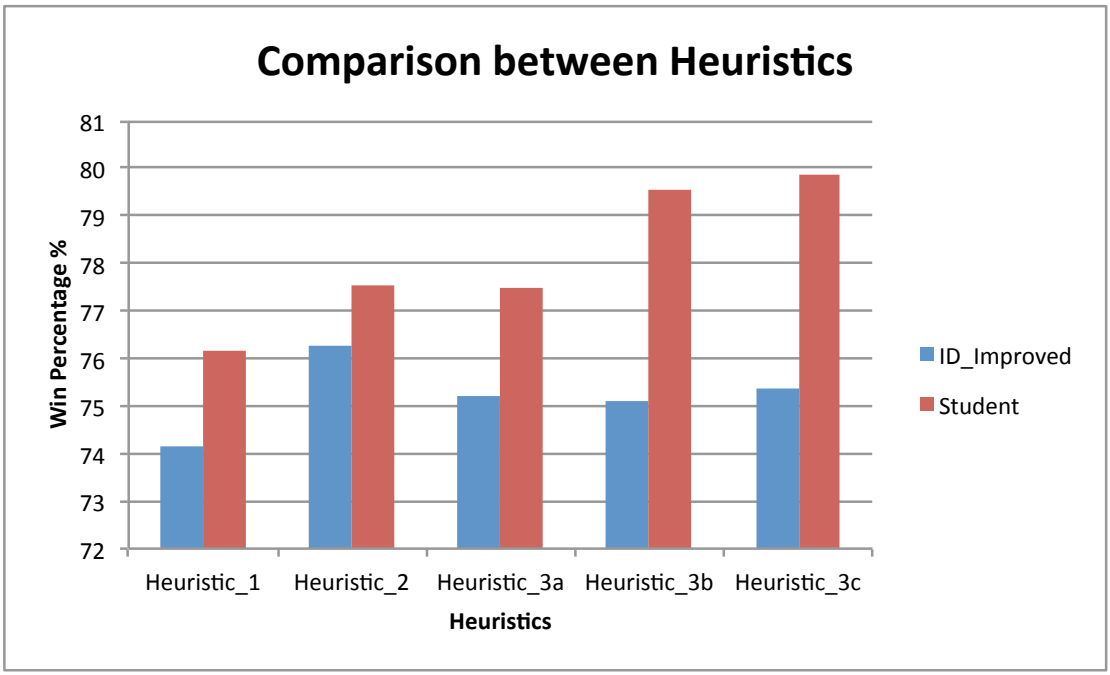


Fig. 11: Chart comparing all the heuristics.

As we can see, the Heuristic 3c performed better than the rest, so I get_move will call this function in the file game_agent.py. The result of the Heuristic_3c was:

```
*************************
 Evaluating: ID  Improved
*************************

Playing Matches:
----------
  Match 1: ID_Improved vs   Random            Result: 378 to 22
  Match 2: ID_Improved vs   MM_Null           Result: 372 to 28
  Match 3: ID_Improved vs   MM_Open           Result: 273 to 127
  Match 4: ID_Improved vs MM_Improved         Result: 274 to 126
  Match 5: ID_Improved vs   AB_Null           Result: 334 to 66
  Match 6: ID_Improved vs   AB_Open           Result: 249 to 151
  Match 7: ID  Improved vs AB  Improved       Result: 230 to 170


Results:
----------
ID_Improved        75.36%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random        Result: 383 to 17
  Match 2:   Student   vs   MM_Null       Result: 381 to 19
  Match 3:   Student   vs   MM_Open       Result: 310 to 90
  Match 4:   Student   vs MM_Improved             Result: 291 to 109
  Match 5:   Student   vs   AB_Null       Result: 358 to 42
  Match 6:   Student   vs   AB_Open       Result: 263 to 137
  Match 7:   Student   vs AB_Improved  Result: 250 to 150


Results:
----------
Student           79.86%
```