heuristic_analysis.md

Introduction

ID Improved is an interesting evaluation function which uses the available moves from both players. It denotes a moderately aggressive play, trying to maximize available moves from the AI while reducing the number of available moves from the opponent.

The result is not that bad, and I feel from the beginning that I'll have to try hard to find a better AI.

I've taken a paper and a pen, played manually, trying to understand the "hidden rules" behind this original game: The moves available are the ones a knight can do in Chess game. here is the list I've created after this analysis, containing the parameters I felt they could impact the success of my AI:

- Number of available moves for the AI
- Number of available moves for the opponent
- The location of particular moves: It seemed to me that we should avoid as much as possible the edges of the game board
- The distance of a move from the edge of the game board (similar to the previous observation but seen as continuous behavior instead of a precise area
- The relative distance between both players: If I can follow my opponent being one square away from him, it seems I can avoid his attacks more easily
- The behavior I should have during the game: Agressive or defensive ? Does it have to change during the game ?
- Following the previous sentence, Should I try to take a square when the opponent has this square as option (attack) or should I avoid it ?

After many "heuristic tentatives", playing with paramaters, I ended up with 3 heuristics trying to find a way to beat the ID improved heuristic:

- Heuristic 1: Aggressive player
- Heuristic 2: Moderately aggressive player trying to keep the center
- Heuristic 3: Aggressive player While 60% of the game board is free, then becoming less aggressive but trying to keep the center.

Heuristic 1: Aggressive player

Description

This one is directly inspired from the lecture: We'll be really aggressive using a multiplier to take the opponent available moves into account.

```
player_moves = len(game.get_legal_moves(player))
opponent_moves = len(game.get_legal_moves(game.get_opponent(player)))
return float(player_moves) - 4 * float(opponent_moves)
```

Results

Data

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

Playing Matches:
----------
tournament.py:100: UserWarning: One or more agents lost a match this round due to timeout. The get_move() 
```

```
  warnings.warn(TIMEOUT_WARNING)
  Match 1: ID_Improved vs   Random      Result: 84 to 16
  Match 2: ID_Improved vs   MM_Null     Result: 71 to 29
  Match 3: ID_Improved vs   MM_Open     Result: 62 to 38
  Match 4: ID_Improved vs MM_Improved   Result: 63 to 37
  Match 5: ID_Improved vs   AB_Null     Result: 68 to 32
  Match 6: ID_Improved vs   AB_Open     Result: 60 to 40
  Match 7: ID_Improved vs AB_Improved   Result: 70 to 30


Results:
----------
ID_Improved         68.29%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random      Result: 89 to 11
  Match 2:   Student   vs   MM_Null     Result: 80 to 20
  Match 3:   Student   vs   MM_Open     Result: 71 to 29
  Match 4:   Student   vs MM_Improved   Result: 57 to 43
  Match 5:   Student   vs   AB_Null     Result: 74 to 26
  Match 6:   Student   vs   AB_Open     Result: 67 to 33
  Match 7:   Student   vs AB_Improved   Result: 67 to 33


Results:
----------
Student             72.14%
```

Analysis

The results have been tested 2 times on tournaments of 100 games. It shows an average improvement of 4% over ID_Improved. I felt being aggressive in this game was important and it's the case: Trying to select the maximum number of possible moves while reducing as much as possible the moves of the opponent is the road to success.

Heuristic 2: Aggressive player trying to keep the center

Description

I refrain a little bit my aggressivity (the multiplier is reduced to 1), but this time I take into account the 9 squares composing the center of the board. And I will privilege these moves for the player while trying to reduce the center moves from my opponent.

```
center_x = game.width//2
center_y = game.height//2
center_locations = [(x, y) for x in [center_x - 1, center_x, center_x + 1] for y in [center_y - 1, center_y

player_moves = len(game.get_legal_moves(player))
opponent_moves = len(game.get_legal_moves(game.get_opponent(player)))

player_center_moves = len([m for m in game.get_legal_moves(player) if m in center_locations])
opponent_center_moves = len([m for m in game.get_legal_moves(game.get_opponent(player)) if m in center_loca

return ((player_moves + player_center_moves) - (opponent_moves + opponent_center_moves))
```

Results

data

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

Playing Matches:
```

```
----------
  Match 1: ID_Improved vs   Random      Result: 90 to 10
  Match 2: ID_Improved vs   MM_Null     Result: 74 to 26
  Match 3: ID_Improved vs   MM_Open     Result: 62 to 38
tournament.py:100: UserWarning: One or more agents lost a match this round due to timeout. The get_move() f
  warnings.warn(TIMEOUT_WARNING)
  Match 4: ID_Improved vs MM_Improved   Result: 63 to 37
  Match 5: ID_Improved vs   AB_Null     Result: 69 to 31
  Match 6: ID_Improved vs   AB_Open     Result: 59 to 41
  Match 7: ID_Improved vs AB_Improved   Result: 55 to 45


Results:
----------
ID_Improved         67.43%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random      Result: 76 to 24
  Match 2:   Student   vs   MM_Null     Result: 81 to 19
  Match 3:   Student   vs   MM_Open     Result: 62 to 38
  Match 4:   Student   vs MM_Improved   Result: 64 to 36
  Match 5:   Student   vs   AB_Null     Result: 75 to 25
  Match 6:   Student   vs   AB_Open     Result: 67 to 33
  Match 7:   Student   vs AB_Improved   Result: 66 to 34


Results:
----------
Student             70.14%
```

Analysis

The results have been tested 2 times on tournaments of 100 games. They show an average improvement of less than 3% compared to the improved heuristic. It seems that I'm on the right way: Being moderately aggressive while taking the center of the board into account has a positive impact on my AI success, which seems to me quite logical, as these squares are the ones giving the highest number of options for the future moves. But it remains a little bit less than the aggressive option. Maybe Should I be more aggressive while keeping this center option...

Heuristic 3: Aggressive player while 60% of the game board is free, then becoming less aggressive keeping the center

Description

I think that the strategy should evolve during the game... with the feeling that aggressivity is key during the first part of the game, then trying to keep the center during the second part of the game, while being less aggressive.

```
def adaptive(game, player):
    board_size = game.height * game.width
    moves_to_board = game.move_count / board_size

    if moves_to_board > 0.4:
        return aggressive(game, player)
    else:
        return aggressive_center(game, player)
```

Results

Data

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

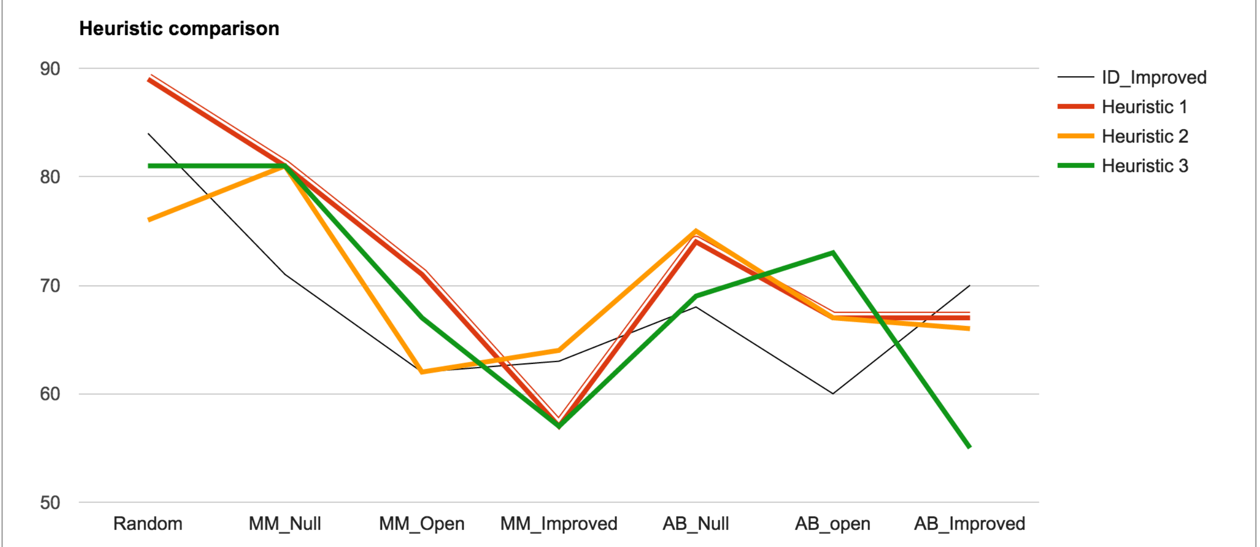
```
*************************

Playing Matches:
----------
  Match 1: ID_Improved vs   Random      Result: 81 to 19
tournament.py:100: UserWarning: One or more agents lost a match this round due to timeout. The get_move() f
  warnings.warn(TIMEOUT_WARNING)
  Match 2: ID_Improved vs   MM_Null     Result: 75 to 25
  Match 3: ID_Improved vs   MM_Open     Result: 64 to 36
  Match 4: ID_Improved vs MM_Improved   Result: 59 to 41
  Match 5: ID_Improved vs   AB_Null     Result: 76 to 24
  Match 6: ID_Improved vs   AB_Open     Result: 65 to 35
  Match 7: ID_Improved vs AB_Improved   Result: 71 to 29


Results:
----------
ID_Improved         70.14%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random      Result: 81 to 19
  Match 2:   Student   vs   MM_Null     Result: 81 to 19
  Match 3:   Student   vs   MM_Open     Result: 67 to 33
  Match 4:   Student   vs MM_Improved   Result: 57 to 43
  Match 5:   Student   vs   AB_Null     Result: 69 to 31
  Match 6:   Student   vs   AB_Open     Result: 73 to 27
  Match 7:   Student   vs AB_Improved   Result: 55 to 45


Results:
----------
Student             69.00%
```

Analysis

The results have been tested 2 times on tournaments of 100 games. I'm quite disappointed with these results. But thinking about it, It seems eventually logical: Keeping the center should be more important at the beginning of the game... But I feel creating an adaptive strategy is a good idea. I'll have to further study this option.

Conclusion / next steps

The aggressive player is the most successfull game-playing agent. Heuristic 1 is the winner.



As explained in my analysis, it seems logical to me as optimizing the number of my moves while reducing as much as possible the number of my opponent's move leads to less options to my opponent.

I'm quite happy with the results, but I feel I could go much further. I think that my AI should adapt its strategy during the game. For example, I thought that following my opponent's path being close to him (1 square away from him) was a good option as a first player during the first part of the game. But my first tries haven't been successfull... Another feeling concerns the symmetries of the game. I've not done anything with it, but I think I could be more efficient taking that into account.

I'm also quite disappointed by the Heuristic 2 results. I thought that these center squares have the highest number of future options, so optimizing these options for me while trying to reduce it for my opponent was a good starting point... Maybe I should try different multipliers.

Last but not least, I should tweak the multipliers and levels to find the most accurate results.

The next steps will be dedicated to testing these parameters and their coefficients. Another interesting topic being to try optimizing the code to go faster (then deeper in the levels then being more confident with the results) in the iterative deepening search.

Great experience anyway. And a new work in progress with the objective of improving this AI.