GameAgentHeuristicAnalysis.md

Project 2 - Game-Playing Agent Heuristic Analysis

Introduction

While working on this project, I implemented a ton of heuristic functions, along with different variations for each of them, using weights and conditionals. I chose only a handful of them to present in this analysis. Performance based on winning ration is presented at the end, in the summary table.

Heuristic 1 - `avoid_edges`

One of the heuristics that I choose to present first in this analysis is the `avoid_edges` one. The idea behind it is to not only count the number of moves that both our agent and its opponent have, but also split them into favorable or not favorable moves. Therefore, we want a function that is maximized when we maximize the favorable moves and minimize the not favorable ones. We note the following heuristic for this variation of the Isolation game (this will be used in later heuristic functions examples too):

> The maximum number of moves that a position could have is 8.

Moreover, the edges and corners seem to be not so ideal positions, as they have 4 and respectively 2 number of moves available from them, in a best case scenario. Therefore we define the `avoid_edges` heuristic as follows:

```
avoid_edges(position) = (own_center_moves + opp_corner_moves) + (own_edge_moves + opp_edge_moves) - (own_corner_moves
```

We want to maximize our center moves and (possibly) edge moves along with the opponent's corner moves and (possibly) edge moves, while minimizing our corner moves and the opponent's center moves.

Heuristic 2 - weighted `avoid_edges`

Since the `avoid_edges` strategy seemed to make sense (in my head at least), I started playing around with weighted versions of it, and got to the following form, which brought some improvements to the original one:

```
avoid_edges(position) = 2 * (own_center_moves + opp_corner_moves) + (own_edge_moves + opp_edge_moves) - 3 * (own_corr
```

This version is more aggressive on minimizing our agent's corner moves and our opponent's center moves, while treating as second priority maximizing its center moves and the opponent's corner moves.

Heuristic 3 - `conditional_score2`

I really wanted to use a conditional approach because I have seen multiple chess agents do something similar - split the game in multiple phases. Also, I have played a lot with my previously presented heuristics + others in order to come to (what seem to be) the right branches of the conditionals, but here the main idea behind it is that:

- At the beginning of the game, the agent does not care that much about the opponent, and is just trying to maximize its own number of moves (getting closer to the idea of a Knight's Tour problem)
- During mid game, we want to maximize the ratio between number of the agent's moves to the board size - we do this in order to really quantify what the count actually means in regards to the game size (something to note here, as stated previously, is that the max move count will be *8*). Moreover, we want to minimize opponent's moves, so the mid game branch also takes care of that.

- At the end of the game we try to maximize our move count as well as minimize the difference between the opponent's move count from the previous position to the current one - this is a somewhat aggressive tactic in which we would try to take spots from the board where the opponent could move towards.

Conclusion

To conclude, I would suggest picking the 3rd heuristic presented here, `conditional_score2` for the following reasons:

- As shown in the Summary Table below, it provides the best average Overall winning rate
- As described above, in the section corresponding to *Heuristic 3*, it looks at both players, but also at the current state of the board, trying to maximize for the positive moves of our agent and minimize the positive moves of the opponent
- A short analysis has been done on the search depth achieved through all of these heuristics, and there was no big difference observed. The average depths, per game phase were:
  - early-game: ~6-7
  - late-game: ~1500

Summary Table

Agent\Heuristic	ID_Improved	avoid_edges	weighted avoid_edges	conditional_score2
Random	95%	95%	100%	95%
MM_Null	90%	85%	90%	90%
MM_Open	65%	75%	80%	90%
MM_Improved	65%	65%	70%	60%
AB_Null	90%	85%	90%	90%
AB_Open	50%	55%	50%	65%
AB_Improved	65%	55%	55%	65%
Overall	74.29%	73.57%	76.43%	79.29%