Nick Garvey
Udacity AIND Build a Game-Playing Agent
Heuristic Report

Below are the results of the three heuristics:

```
(aind) ngarvey@challenger:~/ai/AIND-Isolation$ 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   AB_Custom   AB_Custom_2  AB_Custom_3
                        Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1       Random       8  |   2    10  |   0     9  |   1    10  |   0
    2       MM_Open      6  |   4     5  |   5     7  |   3     8  |   2
    3      MM_Center     8  |   2     7  |   3     9  |   1    10  |   0
    4     MM_Improved    8  |   2     5  |   5     4  |   6     7  |   3
    5       AB_Open      6  |   4     4  |   6     5  |   5     4  |   6
    6      AB_Center     6  |   4     5  |   5     3  |   7     4  |   6
    7     AB_Improved    5  |   5     4  |   6     3  |   7     2  |   8
----------------------------------------------------------------------------
           Win Rate:      67.1%        57.1%        57.1%        64.3%

Your ID search forfeited 84.0 games while there were still legal moves
available to play.
```

As you can see, despite several attempts, I wasn't able to beat the Improved Heuristic in a 1v1 match. Several of my algorithms did beat the Improved Heuristic against weaker opponents however.

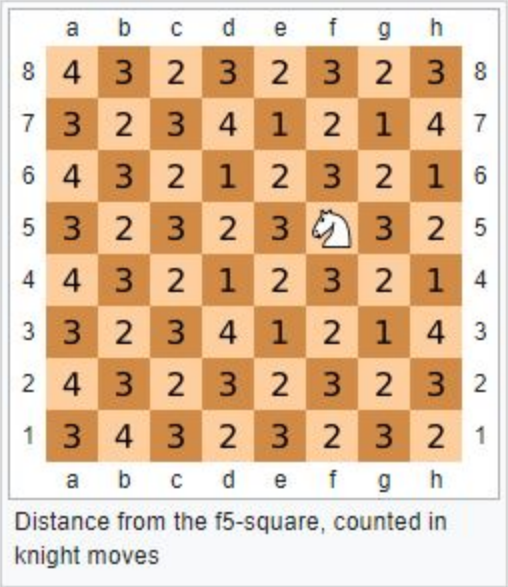
AB_Custom : Many-one-none modified improved score

This heuristic is identical to the improved score heuristic, but it scores the number of open moves as either "more than one available", "just one available" or "none available", instead of differentiating between 2 moves available and 5 moves available.

This performed only slightly worse than AB_Improved against most opponents. It did however perform much worse against the MM_Improved algorithm than AB_Improved did (5|5 vs 8|2). I don't have a good explanation for this, more runs are necessary in order to rule out some unlucky games.

AB_Custom_2 : Open squares of same color

It's generally quite easy to get to squares of the same color, see this image from [2]. Notice how most white squares are only 2 moves away.



Distance from the f5-square, counted in knight moves

This suggests that the number of open squares of the same color could be a good heuristic. I tried it, and it performed okay but not great. It's quite possible that the is_winner and is_loser check did the majority of the heavy lifting here.

AB_Custom_3 : Opens squares away in three moves
This counts the squares open in three moves and uses that as the heuristic. It doesn't consider counter attacks - that would be a waste given we'd rather just iterate deeper in that case.

This performed quite well against most opponents, but got demolished by AB_Improved. The lack of counter attack consideration was likely the reason here. A better heuristic could be combining the number of open squares 3 away with the number of open squares 1 away. If there's only one square open nearby, then this should bring down the heuristic score significantly.

Conclusion & Three Reasons for Heuristic Choice

I spent a fair amount of time trying to beat improved score, but wasn't successful. I think this is mostly due to my own lack of skill at this game - I don't have a good personal heuristic so I can't come up with a good one computationally.

I ended up picking the heuristic that performed best against AB_Improved as AB_Custom, even though other heuristics did better against most opponents. It performs well against the other heuristics as well making it a good choice. It also performs quite well against the random player.

There is a lot of research on Knight Tours, and a good heuristic there is Warnsdorf's Rule [1]. Warnsdorf's Rule simply dictates to pick the move with the least number of options. This ends up working out well in a non-competitive case, as you leave a lot of options open as you progress. This obviously doesn't work well in the competitive case as your opponent can simply attack you.

[1] https://en.wikipedia.org/wiki/Knight%27s_tour#Warnsdorf.27s_rule
[2] https://en.wikipedia.org/wiki/Knight_(chess)