Heuristic analysis

Figure : The agents played a tournament of 80 matches, with 150 ms allowed for each move (NUM\_MATCHES = 80, TIME\_LIMIT = 150)

 Match #   Opponent    AB\_Improved   AB\_Custom   AB\_Custom\_2  AB\_Custom\_3

                        Won | Lost   Won | Lost   Won | Lost   Won | Lost

    1       Random      148 |  12    148 |  12    149 |  11    139 |  21

    2       MM\_Open     117 |  43    110 |  50    114 |  46    111 |  49

    3      MM\_Center    139 |  21    143 |  17    136 |  24    137 |  23

    4     MM\_Improved   118 |  42    113 |  47    100 |  60    97  |  63

    5       AB\_Open     85  |  75    85  |  75    73  |  87    75  |  85

    6      AB\_Center    88  |  72    94  |  66    87  |  73    75  |  85

    7     AB\_Improved   79  |  81    74  |  86    69  |  91    67  |  93

--------------------------------------------------------------------------

           Win Rate:      69.1%        68.5%        65.0%        62.6%

# Introduction

From earlier tournaments, I found that the AlphaBeta (AB) agents decisively beat their MiniMax (MM) counterparts. The AB agents are able to search more deeply in the given time, because they can prune branches. So I surmise that searching deeply gives a strong advantage in this game.

Thus I tried to develop heuristics which would run quickly, so that my agents could search more deeply. This meant I focused on features of the board that I could examine directly, rather than computing features.

# AB\_Custom: Player separation

This was my best heuristic: the (squared) distance across the board between the players. Its performance in the tournament was on par with AB\_Improved.

I favour this heuristic for three reasons:

1. *I can rationalise it.* It makes sense to stay away from your opponent, as it makes it harder for them to interfere with you, i.e. blocking your moves.
2. *It is fast.* We require only five operations (two subtractions, two multiplications, one addition).
3. *It performs well across all opponents in the tournament.* This gives me hope that it will beat yet unseen agents.

# AB\_Custom\_2: Row/column matching

This heuristic gives a positive signal if I am on the same row or column of the board as my opponent. Otherwise it returns zero.

It is the fastest heuristic, requiring two comparisons only. However, it seems fairly arbitrary; it’s not obvious what strategy this executes.

Row/column matching does notably well against the Center heuristic. This probably has something to do with the patterns of knight movement on the board.

# AB\_Custom\_3: Opponent blocking

My last heuristic returns a positive signal if I am blocking one of my opponent’s otherwise legal moves. Otherwise it returns zero. The rationale is to interfere with my opponent and impede their strategy.

Unfortunately this heuristic didn’t manage to beat any of the other AB agents. This could be because it is slow to run, limiting the depth to which the agent can search.

# Discussion

AB\_Custom (player separation) was my best heuristic, and performed on par with AB\_Improved, the benchmark.

However, given a choice between all four agents, I would pick AB\_Improved. This is because it has the clearest rationale out of all of the heuristics. I can imagine boards on which the best move would be to move *towards* my opponent, because doing otherwise would limit my moves and give the opponent more freedom. Only the AB\_Improved heuristic would guide me in these circumstances.

Even though their performance was very close (perhaps indistinguishable under a formal statistical analysis), I would bet on AB\_Improved beating a greater number of yet unseen agents in other tournaments.

# Future work

There is a lot more depth to explore (excuse the pun) in this project. One thing that slowed this effort was the long time required to run each tournament.

A vital next step would be to demonstrate whether the differences in performance between the agents is statistically significant. Especially given how closely matched my best heuristic appears to be with the benchmark.

I would like to verify my assumption that searching deeply, i.e. quickly, gives a strong advantage. This could be achieved by comparing the performance of agents restricted to different fixed depths.

Finally, each of my agents utilises only a single heuristic. I imagine a champion agent would combine a number of different heuristics when evaluating a board.