The results of tournament.py are recorded below:

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 9 | 1 8 | 2 7 | 3 8 | 2
2. MM\_Open 6 | 4 6 | 4 7 | 3 6 | 4
3. MM\_Center 7 | 3 7 | 3 9 | 1 6 | 4
4. MM\_Improved 5 | 5 9 | 1 8 | 2 4 | 6
5. AB\_Open 5 | 5 5 | 5 6 | 4 6 | 4
6. AB\_Center 5 | 5 6 | 4 6 | 4 4 | 6
7. AB\_Improved 6 | 4 6 | 4 4 | 6 6 | 4

-------------------------------------------------------------------------- Win Rate: 61.4% 67.1% 67.1% 57.1%

There were 27.0 timeouts during the tournament -- make sure your agent handles search timeout correctly, and consider increasing the timeout margin for your agent.

**Analysis**

All of my algorithms ended up being basic riffs off of AB\_Improved and AB\_Center.

AB\_Custom takes AB\_Improved and adds (AB\_Center / 24.5) This regularizes it to a single possible extra move for being further away from the center. The agent consistently beat AB Improved. The results above are consistent across several tournaments I ran.

AB\_Custom\_2 takes AB\_Improved and multiplies opponents\_moves by 1.5, leading to more aggressive play. It then adds (AB\_Center / 24.5) This regularizes it to a single possible extra move for being further away from the center. The agent consistently performed better against AB\_Open and center but lost against AB\_Improved. I ebleive that is because it weighs blocking openents moves too heavily and doesn’t weigh its own move possibilities well enough, leading to too aggressive blocking behavior.

AB\_Custom takes AB\_Improved and adds 2 \* (AB\_Center / 24.5) This regularizes it to two possible extra moves for being further away from the center. The algorithm consistently beat AB\_Improved, but I think it weighs being away from center too heavily.

**Recommendation**

Across multiple tournaments, the most consistent winner was the simple combination of AB\_Improved with a normalized AB\_Center (my AB\_Custom.) The algorithm is simple to understand and I believe it properly weighs game paths: all things being equal it’ll always choose the path that maximizes its own number of moves and minimizes the opponents number of moves. That said, if multiple paths offer the same move calculus, it’ll weigh the one furthest from center most heavily. All algorithms implemented will never time out without offering a move because of iterative deepening, which I believe is a good strategy.