Heuristic Function Analysis | Adversarial Search

Udacity Artificial Intelligence Nanodegree
Brittany Martin | 12/26/2017

Abstract

We developed an adversarial search agent to play the game "Isolation". Agents have a fixed time limit each turn to search for the best move and respond. We were tasked with creating three of our own heuristics. The final step was to test the relative performance of our agents in a round-robin tournament against several other predefined agents.

Heuristics

- *Custom Score 1 (Available Moves for Active Player)* - make decisions solely based on the number of available moves for the active player.

- *Custom Score 2 (Weighted Difference - Attempt 1)* - finds the difference between the the active player's moves and the weighted opponent's moves to locate next best move. The opponent is weighted by 2.

- *Custom Score 3 (Weighted Difference - Attempt 2)* - finds the difference between the the active player's moves and the weighed opponent's moves to locate next best move. The opponent is weighted by .5.

Hypothesis

I would expect out of the custom heuristics, Custom Score 1 would perform the worst since it only takes the active player's moves into consideration. I would also assume that Custom Score 2 would perform the best.

I hypothesize all of my custom functions to outperform Random but to be beaten by AB_Improved. Custom Score 2 and Custom Score 3 should have an upperhand on all MM challengers since they utilize alpha-beta pruning.

Results

```
                      ************************
                          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      8 |  2      6 |  4
    2       MM_Open     6 |  4      3 |  7      8 |  2      6 |  4
    3       MM_Center   8 |  2      5 |  5     10 |  0      9 |  1
    4       MM_Improved 6 |  4      5 |  5      7 |  3      8 |  2
    5       AB_Open     6 |  4      3 |  7      6 |  4      5 |  5
    6       AB_Center   5 |  5      7 |  3      6 |  4      6 |  4
    7       AB_Improved 3 |  7      5 |  5      2 |  8      5 |  5
--------------------------------------------------------------------------
            Win Rate:      61.4%        51.4%        67.1%        64.3%
```

Figure 1. Tournament Results

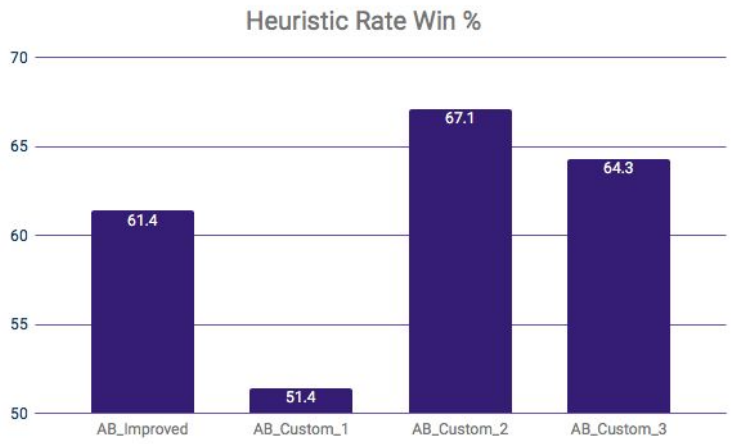

Figure 2. Win Rate % for Tournament

As you can see from Figure 1 and 2, the hypotheses matched the following:

1. All three custom score functions consistently beat the Random player.
2. Custom Score 2 was the most performant at 67.1%, beating AB_Improved at 61.4%.
3. Custom Score 2 and 3 easily beat the MM challengers. Custom Score 2 even beat MM_Center 100% of the time.

Analysis

This tournament proved that Alpha-Beta pruning is a reliable optimization technique for minimax algorithm since it allowed the algorithm to search faster and go into deeper levels in the gametree. Custom Score 2 is preferred algorithm.