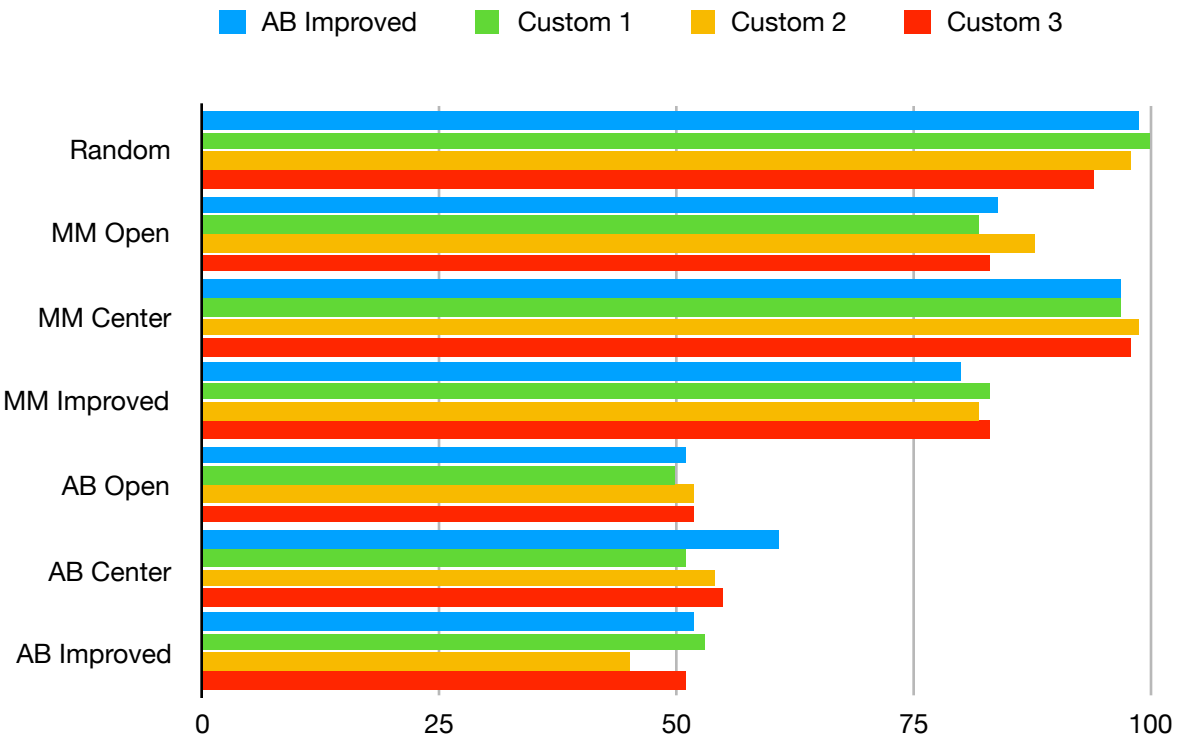
Chad Iverson

Artificial Intelligence Nanodegree Program

December 31, 2017

Heuristic Selection for Agent Playing Game of Isolation

Agents using three experimental evaluation functions barely marked an improvement over the performance of agents using pre-built functions. As expected, agents that employed alpha–beta pruning, with iterative deepening, won a majority matches against the minimax agents across all functions with a win rate ranging from 82% to 99%. However, when agents in the alpha-beta class faced off against each other, performance improvements attributable to custom evaluation functions appeared to offer no statistically significant advantage, at the alpha level of .05 or .10. The observed advantage over AB Center was slightly higher for all custom functions. An advantage over the pre-built AB Improved was only observed for functions Custom 1 and Custom 3 with agents winning 53 and 51 times respectively out of 100 total matches.



Match #	Opponent	AB Improved		AB Custom		AB Custom 2		AB Custom 3	
		Won	Lost	Won	Lost	Won	Lost	Won	Lost
1	Random	99	1	100	0	98	2	94	6
2	MM Open	84	16	82	18	88	12	83	17
3	MM Center	97	3	97	3	99	1	98	2
4	MM Improved	80	20	83	17	82	18	83	17
5	AB Open	51	49	50	50	52	48	52	48
6	AB Center	61	39	51	49	54	46	55	45
7	AB Improved	52	48	53	47	45	55	51	49
	Win Rate	74.9%		73.7%		74.0%		73.7%	

Each custom evaluation function used newly created heuristics for game state evaluation based on player intuition while extending pre-built features. The first custom function identified when a player is in one of two positions, relative to its opponent, where it can draw the opponent to the left side of the board. In cases where the player is not in a position to draw the opponent, the pre-built “improved” heuristic is used to score the game state. The idea behind drawing an opponent to the edge of the board is meant to accelerate isolation. The second function used a proximity heuristic to measured relative distance, scoring higher for moves closer to the opponent. It also incorporates a weighted version of the pre-built “improved” heuristic that scores using moves available to the current player minus moves available to the opponent. The third, and last custom evaluation function returned higher scores for moves into less crowded quadrants of the board and also incorporated the “improved” heuristic.

Based on the tournament results, presented in the chart above, I recommend function Custom 1. When comparing custom evaluation functions, the numbers are inconclusive. That said, the first reason for supporting function Custom 1 is its win rate

against the baseline AB Improved. None of the custom functions had a better win rate than the baseline function; however, function Custom 1 came in on top, with a win rate of 53% against AB Improved. The second reason for recommending Custom 1 is that while achieving a slight edge over the baseline function, the results did not reveal any disadvantage for Custom 1 when facing off against agents using other heuristics. At its worst, Custom 1 had a 50% win rate against AB Open. At its best, Custom 1 had a 100% win rate against Random making it the only undefeated agent in the first match. The third reason for recommending Custom 1 is how it treats the opponent when compared to the other functions. Custom 1 specifically takes advantage of knowing that the baseline function employs the improved heuristic. It targets positions, according to how legal moves are determined, that will cause this heuristic to draw the opponent toward the edge of the board. The fact that Custom 1 incorporated this strategy, while still achieving an overall win rate just 1.2% under the overall win rate for AB Improved is promising. This approach to heuristic design holds potential for future improvements considering both game-specific and gameplay knowledge.

Looking at the agent program overall, I recommend using heuristics to reduce the search space further. The AB agents employed iterative deepening and a first level cache. Still, testing revealed timeouts consistently for the first eight moves, even after extending the timeout by more than 60 times. Selective reduction of the search space should be the next focus area. Consider moving beyond the AB implementation and onto more sophisticated heuristic search algorithms.