Artificial Intelligence Nanodegree
Udacity

July 11, 2017

Heuristic Analysis

Heuristic-1

The **aggressive_heuristic** is based on the fact that heuristic could be made more aggressive by assigning a higher cost to remaining opponent moves. Accourding to this heuristic, agent prefers moves that reduce the options available to its opponent. This heuristic is calculated as

$$player\_moves - (w * opponent\_moves) \tag{1}$$

where $w > 1$.

Heuristic-2

aggressive_move_count_heuristic is a special use of **aggressive_heuristic** that assigns a higher value to moves as the game progresses. As the number of options increase as the game progresses, it is better to consider a move that provides a positional advantage later in the game is worth more than a similar advantage earlier in the game. This heuristic is calculated as

$$move\_count * (player\_moves(w * opponent\_moves)) \tag{2}$$

where $w = 3$.

Heuristic-3

The **reachable_squares_heuristic** attempts to look ahead several turns in addition to the number of moves left to each player. For each player, it determines the set of squares reachable from the current square, then the squares reachable from those, and so on, recursively to depth d. The number of unique squares in the set is that player's component of the score. The final heuristic value is then calculated as

$$reachable\_squares(player)w * reachable\_squares(opponent) \quad (3)$$

where w is a positive integer weight, as in the other heuristics. According to the test, better choice are $d = 3$ and $w = 2$.

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	18	2	18	2	17	3	18	2
2	MM_Open	16	4	18	2	15	5	12	8
3	MM_Center	18	2	17	3	18	2	20	0
4	MM_Improved	15	5	15	5	15	5	13	7
5	AB_Open	10	10	10	10	12	8	12	8
6	AB_Center	14	6	14	6	12	8	12	8
7	AB_Improved	9	11	12	8	8	12	8	12
	Win Rate:	71.4%		74.3%		69.3%		67.9%	

The output of 10 game played with different heuristics are shown in the table above. The custom score functions are:

Label	Heuristic function
AB_Custom	reachable_squares_heuristic
AB_Custom_2	aggressive_heuristic
AB_Custom_3	aggressive_move_count_heuristic

Overall, **reachable_squares_heuristic** had the best win rate, followed by **aggressive_move_count_heuristic** , **aggressive_heuristic** , and finally the original "improved" heuristic.

The **reachable_squares_heuristic** (*AB_Custom* in the results table) is the recommended score function. It consistently achieved the best overall win rate and showed relative strength across all opponent types. It does require a little work than the other heuristics (3x as many calls to **Board.__get_moves** plus 3 hashset unions), but the implementation is quite simple and still relatively quick. The

heuristic also makes intuitive sense; an advantage in available moves that endures through several turns suggests a significant positional advantage.