Heuristic Analysis

Jinlin Song

Udacity AIND

Build a Game-Playing Agent

Evaluation Functions

In order to win the isolation game. Intuitively, more valid moves means high probability to win the game, so that our algorithm should make sure we have as much moves as possible, and meanwhile, our opponent has as little moves as possible, though it can't guarantee we can win in the final. I put forward three different heuristic functions.

custom_score:

(owner valid moves + next round owner valid moves) - (opponent valid moves + next round opponent valid moves);

custom_score_2:

owner valid moves - opponent valid moves;

custom_score_3:

*owner valid moves - 2 * opponent valid moves.*

Performance of Agents Against the Testing Agents

Agents Analyse

Instance 1:

```
*************************
      Playing Matches
*************************
```

Match #	Opponent	AB_Improved Won	Lost	AB_Custom Won	Lost	AB_Custom_2 Won	Lost	AB_Custom_3 Won	Lost
1	Random	9	1	10	0	9	1	9	1
2	MM_Open	8	2	8	2	7	3	9	1
3	MM_Center	9	1	9	1	9	1	8	2
4	MM_Improved	8	2	6	4	8	2	9	1
5	AB_Open	6	4	5	5	6	4	5	5
6	AB_Center	5	5	5	5	5	5	8	2
7	AB_Improved	4	6	5	5	5	5	6	4
	Win Rate:	70.0%		68.6%		70.0%		77.1%	

Figure 1

Instance 2:

```
*************************
      Playing Matches
*************************
```

Match #	Opponent	AB_Improved Won	Lost	AB_Custom Won	Lost	AB_Custom_2 Won	Lost	AB_Custom_3 Won	Lost
1	Random	10	0	9	1	10	0	10	0
2	MM_Open	8	2	10	0	8	2	8	2
3	MM_Center	9	1	8	2	10	0	9	1
4	MM_Improved	9	1	6	4	6	4	7	3
5	AB_Open	5	5	5	5	5	5	5	5
6	AB_Center	7	3	6	4	6	4	4	6
7	AB_Improved	4	6	5	5	7	3	7	3
	Win Rate:	74.3%		70.0%		74.3%		71.4%	

Figure 2

Instance 3:

```
                        ************************
                            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       10  |  0       10  |  0       10  |  0
   2        MM_Open      8  |  2        7  |  3        7  |  3       10  |  0
   3       MM_Center     9  |  1       10  |  0       10  |  0        8  |  2
   4      MM_Improved    9  |  1       10  |  0        8  |  2        9  |  1
   5        AB_Open      4  |  6        5  |  5        6  |  4        7  |  3
   6       AB_Center     7  |  3        4  |  6        6  |  4        8  |  2
   7      AB_Improved    6  |  4        5  |  5        6  |  4        4  |  6
         ------------------------------------------------------------------------
           Win Rate:      74.3%          72.9%          75.7%          80.0%
```

Figure 3

From Figure 1, Figure 2 and Figure 3, all heuristic functions' win rates are around 70%. Max win rate is 80%, and minimum win rate is 68.6%. None of their win rates are consistent. That is to say, their performances are highly related to their opponents, moreover it vary from time to time even under the same opponent. The reason is the process of searching the best action is not based on a complete game tree and the heuristic function is not equal to win rate.

Testing Agents Analyse

Opponent	custom_score	custom_score_2	custom_score_3
Randon	96.67%	96.67%	96.67%
MM_Open	83.33%	73.33%	90.00%
MM_Center	90.00%	96.67%	83.33%
MM_Improved	73.33%	73.33%	83.33%
AB_Open	50.00%	56.67%	56.67%
AB_Center	50.00%	56.67%	66.67%
AB_Improved	50.00%	60.00%	56.67%

Table 1 (Overall win rates for each heuristic function under different opponents, game times = 30, including Instance 1, 2 and 3)

From Table 1, we observed when compete with "Random", every heuristic function have 96.67% win rate. What is more, our heuristic functions still perform pretty awesome under MM_Open, MM_Center and MM_Improved, it can have 96.67% win rate, with minimum win rate 73.33%. However when we compete with AB_open, AB_center and AB_Improved, win rate will drop down to 50% for custom_score, and the best overall win rate is just a little above 50%, 66.67%, that is for custom_score_3, when it compete with opponent AB_Center. From "Table 1", we also conclude that AB_Imporved is the strongest opponent under our evaluation functions.

Recommendation about the best evaluation function:

The reason I select custom_score as my final heuristic function is that custom_score is more stable than the rest heuristic functions. First of all, its results don't vary a lot from time to time. Custom_score is more reliable even its overall win rate is below than custom_score_2 and custom_score_3. Secondly, custom_score's win rate is above 70%, which is pretty close to her peers, thought it doesn't beat other heuristic function under some cases. Finally, custom_score take next round's valid move numbers into consideration, which is further than custom_score_2 and custom_score_3.