Udacity AIND-Build an Adversarial Game Playing Agent

Tianwei LAN

Option 1 chosen: Develop a custom heuristic

Performance baseline table (with the original heuristic = #my_moves - #opponent_moves):

Opponent	Number of Matches	Time Limit (ms)	Winning Percentage
RANDOM	200	150	92.5%
GREEDY	200	150	84.0%
MINIMAX	200	150	66.5%
SELF	200	150	50.0%

I set common_liberties as the common liberties that both my_moves and opponent_moves have in common.

Then, I created 9 different custom heuristics in order to test if I could find a custom heuristic better than the original heuristic.

Ori = #my_moves - #opponent_moves

Cus_1 = 2*#my_moves - #opponent_moves

Cus_2 = #my_moves - 2*#opponent_moves

Cus_3 = #my_moves - #opponent_moves + #common_liberties

Cus_4 = 2*#my_moves - #opponent_moves + #common_liberties

Cus_5 = #my_moves - 2*#opponent_moves + #common_liberties

Cus_6 = #my_moves - 3*#opponent_moves + #common_liberties

Cus_7 = #my_moves - 2.5*#opponent_moves + #common_liberties

Cus_8 = #my_moves - 2*#opponent_moves + 2*#common_liberties

Cus_9 = #my_moves - 2*#opponent_moves + 1.5*#common_liberties

In order to compare with the original heuristic, for each of these 9 custom heuristics, I ran 200 matches against MINIMAX opponent with 150ms time limit for each action and depth limit as 5 for iterative deepening:

Heuristics	Ori	Cus_1	Cus_2	Cus_3	Cus_4	Cus_5	Cus_6	Cus_7	Cus_8	Cus_9
Winning Percentage	66.5%	62.0%	62.0%	53.5%	57.0%	67.5%	60.5%	66.0%	54.5%	73.0%

As shown in the table above, the winning percentage of Cus_5 (67.5%) was slightly higher than the original one (66.5%). With the intention to increase the winning percentage, I began to tune the coefficient of #common_liberties. Finally, I decided to choose Cus_9 as my custom heuristic because its winning percentage (73.0%) is much higher than the winning percentage of the original heuristic (66.5%).

My final custom heuristic:

Cus_9 = #my_moves - 2*#opponent_moves + 1.5*#common_liberties

Performance table (with Cus_9):

Opponent	Number of Matches	Time Limit (ms)	Winning Percentage
RANDOM	200	150	95.0%
GREEDY	200	150	85.5%
MINIMAX	200	150	73.0%
SELF	200	150	50.0%

Q1: What features of the game does your heuristic incorporate, and why do you think those features matter in evaluating states during the search?

I use the number of the custom agent's own liberties, the number of the opponent's liberties and the number of the common liberties in my custom heuristic. I set common_liberties as the common liberties that both my_moves and opponent_moves have in common. In fact, the custom agent tries to maximize its own number of liberties while minimizing the opponent's liberties. Particularly, the custom agent should pay more attention to the common liberties which can make itself move and decrease the number of liberties for the opponent at the same time. The coefficients for these three terms can describe whether the custom agent should be more aggressive in order to maximize its winning percentage.

Q2: Analyze the search depth your agent achieves using your custom heuristic. Does search speed matter more or less than accuracy to the performance of your heuristic?

Using the custom heuristics Cus_9, I ran 200 matches against MINIMAX opponent with 150ms time limit for each action and depth limit from 1 to 5 for iterative deepening:

Depth Limit	Total Execution Time (s)	Winning Percentage
1	85	25.0%
2	90	25.5%
3	110	52.0%
4	141	54.0%
5	232	73.0%

As the depth limit increases, both the total execution time and the winning percentage rise dramatically. Therefore, there is a trade-off between the search speed and the accuracy to the performance of my heuristic. If I want to have a better accuracy to the performance and a higher winning percentage, I must accept longer search time.