2020

Isolation Game

AI NANODEGREE PROJECT 3 REPORT

GORKEM BUZCU

Table of Contents

Introduction

In this project our task was to complete the code in isolation game and then run matches against different algorithms in order to create a benchmark. I have run all 4 algorithms against my agent. For the second part of my project I chose to implement new heuristics.

Algorithm List

1	Greedy
2	Random
3	Minimax
4	Self

Heuristics List

1	Baseline
2	Weighted Offensive
3	Progressively Defensive
4	Progressively Offensive
5	Defensive to Offensive
6	Weighted Liberty

Results

Unit Test Results

The unit test results of my_custom_player:

root@2ec8e1cb60b3:/home/workspace# python -m unittest -v

test_get_action_midgame (tests.test_my_custom_player.CustomPlayerGetActionTest)

get_action() calls self.queue.put() before timeout in a game in progress ... ok

test_get_action_player1 (tests.test_my_custom_player.CustomPlayerGetActionTest)

get_action() calls self.queue.put() before timeout on an empty board ... ok

test_get_action_player2 (tests.test_my_custom_player.CustomPlayerGetActionTest)

get_action() calls self.queue.put() before timeout as player 2 ... ok

test_get_action_terminal (tests.test_my_custom_player.CustomPlayerGetActionTest)

get_action() calls self.queue.put() before timeout when the game is over ... ok

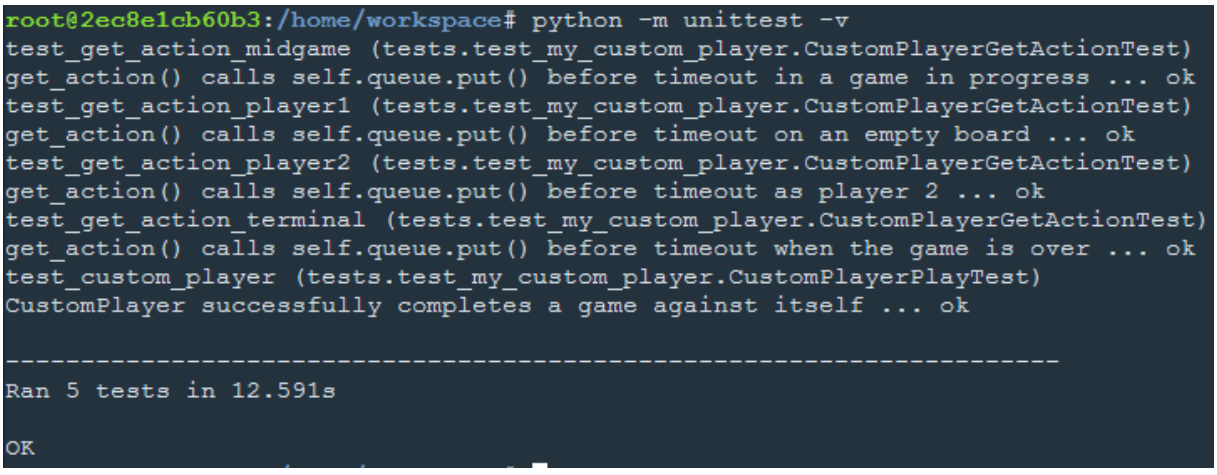
test_custom_player (tests.test_my_custom_player.CustomPlayerPlayTest)

CustomPlayer successfully completes a game against itself ... ok

---

Ran 5 tests in 12.591s

OK



Heuristics Match Results

Baseline Heuristics

This is the simplest heuristic. It just returns the number of liberties for the player minus opponent liberties.

Weighted Offensive

This heuristic returns player's liberties minus opponent's liberties times two.

Progressively Defensive

This heuristic favors more liberty for the player as the game progresses.

Progressively Offensive

This heuristic has a higher penalty from opponent's liberties. As the game progresses and number of ply increases the penalty increases.

Defensive to Offensive

My research about isolation game led me to realize that playing defensive at the beginning and switching to offensive later on the game is a good strategy.

Weighted Liberty

During my research I also learned that controlling center locations is indeed a good strategy. In this heuristic I favor middle locations 3 times more than outer locations.

In the table below you can see match results. The parameters used were –f –r 10 –o algorithm –p 4. These parameters used in each run to assure consistency. In the results below I did not limit the depth to ensure best win ratio.

Win Ratios	Baseline	Weighted Offensive	Progressively Defensive	Progressively Offensive	Defensive to Offensive	Weighted Liberty
Greedy	92,5	82,5	90	97,5	92,5	85
Random	95	95	90	97,5	92,5	90
Minimax	80	70	75	85	75	57,5
Self	55	62,5	47,5	57,5	55	62,5

Table 1 Match Results

In the table below I limited max depth to 4 and got the results.

Win Ratios	Baseline	Weighted Offensive	Progressively Defensive	Progressively Offensive	Defensive to Offensive	Weighted Liberty
Greedy	70	85	75	67,5	80	70
Random	97,5	92,5	92,5	95	97,5	67,5
Minimax	50	40	47,5	45	60	50
Self	50	50	50	50	50	47,5

Table 2 Match Results

In the table below I limited max depth to 16 and got the results.

Win Ratios	Baseline	Weighted Offensive	Progressively Defensive	Progressively Offensive	Defensive to Offensive	Weighted Liberty
Greedy	95	87,5	95	95	97,5	82,5
Random	95	97,5	95	95	95	100
Minimax	85	75	75	77,5	75	65
Self	45	50	57,5	57,5	60	47,5

Table 3 Match Results

Questions and Answers

Q1

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

I will answer this question on my heuristic function called weighted liberty. In this algorithm not all liberties are the same. When I was researching about this game, I came across resources that mention center squares are important in order to have a better end game. That is the reasoning behind scoring center squares three times more than others. But in the end when match results considered this heuristic did not perform well. Also I saw that defensive to offensive method works well. Maybe combining these two functions and favoring center locations at the beginning and then switching to an offensive method will be better.

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?

I have run matches with depth limited to 4, 16, and unlimited. When I limit depth to 4 the runtime decreased dramatically. It was very quick but the results were not promising. When I limit it to 16 the search took lot longer than before but the results were much better. I can't say that speed or accuracy matter more than the other. It is important to find a balance between speed and accuracy. It is better to use as much time as we can in order to help accuracy without losing too much time.