Udacity Artificial Intelligence Degree:
Build a game playing agent

Introduction

This document is part of the Udacity Artificial Intelligence degree program, as part as the solution for the "Build a game playing agent" project.

The project is based in the Isolation game. Isolation is a deterministic, two-player game of perfect information in which the players alternate turns moving a single piece from one cell to another on a board. Whenever either player occupies a cell, that cell becomes blocked for the remainder of the game. The first player with no remaining legal moves loses, and the opponent is declared the winner.

The students are asked to implement three different heuristics that can improve pre-implemented ones.

Heuristic 3 (custom_score_3 function)

The first heuristic was a simple loss that uses the number of moves available for the current player and the opponent.

The idea is that the number of moves is already proven as a good metric, so I added some weights to those two components to try to get to a better result. The weights chosen were 1.5 for the "human" player moves and 1.0 for the opponent ones, therefor giving more importance to the human remaining moves.

Heuristic 2 (custom_score_2 function)

We add here three additional components:

- Distance to the center of the board for the human position (positive metric)
- Distance to the center of the board for the opponent position (negative metric)
- Is the human in a distance=3 from the opponent? (positive metric)

The first two components assume that being close to the center of the board is a good thing, as it allows a bigger freedom of movements. Therefore, it would be something beneficial for the human (positive metric) and vice versa.

The third component is kind of a "boost" factor. If the human is exactly in a distance = 3 from the opponent, not only reduces in one its number if movements, but it should be in a good place to "harass" it. Therefore, in this particular situation, we add some value to the metric.

Heuristic 1 (custom_score function)

The components of this heuristic are the same ones as the heuristic 2. The only thing that changes are the weights assigned to each one of the components.

In order to optimize these weights, we use a bayesian method for parameters optimization called Spearmint, developed by Ryan P. Adams, Michael Gelbart, and Jasper Snoek at Harvard University, Kevin Swersky at the University of Toronto ("Toronto"), and Hugo Larochelle at the Université de Sherbrooke ("Sherbrooke") (https://github.com/HIPS/Spearmint).

We run 590 simulations of the tournament playing against a subset of the opponents (MM_Improved, AB_Open and AB_Improved), and use the total wins percentage as a score metric.

At the end of the simulations, we got the parameters that are currently defined in the custom_score function. Interestingly, some of the weights have a negative value. These values would contradict the original intend of the parameter, but they have been unmodified for the sake of the validity of the method.

Results

These are the final results of the tournament for each one of the heuristics:

Match	Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3	
		Won	Lost	Won	Lost	Won	Lost	Won	Lost
1	Random	19	1	19	1	19	1	19	1
2	MM_Open	14	6	14	6	15	5	15	5
3	MM_Center	18	1	17	3	16	4	14	6
4	MM_Improved	16	4	15	5	16	4	14	6
5	AB_Open	11	9	14	6	11	9	10	10
6	AB_Center	13	7	10	10	11	9	9	11
7	AB_Improved	10	10	9	11	11	9	12	8
	Win rate	**72.1%**		**70.0%**		**70.7%**		**66.4%**	

We can see that both AB_Custom and AB_Custom_2 were very close to AB_Improved performance. This suggest that the chosen strategy can be feasible.
However, despite the results during the parameters optimization were better, the optimized combination of parameters in AB_Custom could not improve the results in AB_Custom_2. This may suggest that the manually estimated weights performed already very good for the chosen strategy.

Final heuristic recommendation

It is hard to make a recommendation between AB_Custom and AB_Custom_2. Even though AB_Custom uses the same parameters and obtained better results during the parameters optimization, it did not show a significant improvement in the tournament's performance over AB_Custom_2. Sometimes one is slightly better than the other, but not with very consistent results. However, both of them perform better than AB_Custom_3, which makes sense given that they follow a similar strategy but they include more parameters to achieve a higher performance.
Also note that both AB_Custom and AB_Custom_2 have a very similar overall performance that AB_Improved, with AB_Custom_2 winning in this particular tournament.
Considering this, I would probably recommend AB_Custom_2 after all. The main reason is that the weights of the parameters make more sense to me than the ones found by the empiric Spearmint approach. During the experiments, I could not find any clear pattern in the parameters when the solution space is big, so I would probably start with AB_Custom_2 and do small variations in the heuristic manually chosen parameters if required.
However, it is also possible that a bigger number of experiments could finally lead to a combination of parameters that makes AB_Custom consistently better than AB_Custom_2, so I don't really see a clear winner between both heuristics.