Heuristics Analysis

Introduction

The goal of this project is to develop an adversarial search agent to play the game "Isolation", where each agent is restricted to L-shaped movements (like a knight in chess) on a rectangular grid (like a chess or checkerboard).

I implemented the following functions:

- implement minimax algorithm for game-tree search.
- implement minimax search with alpha-beta pruning.
- implement iterative deepening search.
- develop 3 heuristic evaluation functions: the goal is to develop a heuristic such that Student outperforms ID_Improved.

I implemented the functions following AIMA pseudocode specifications, as suggested in README.md file, for minimax, alpha-beta, and iterative deepening.

Heuristic Evaluation Functions

I implemented three different heuristics, trying to develop a low cpu-time consuming code, because heuristic validation should be fast:

1. custom_score: the idea is to calculate the Manhattan distance between two points (the player and the opponent) measured along axes at right angles. In a plane with p_1 at (x_1, y_1) and p_2 at (x_2, y_2), it is $|x_1 - x_2| + |y_1 - y_2|$. The score for a position is set as the distance between the player and the opponent.

2. custom_score_2: starting from the *improved_score* heuristic, provided in *sample_players.py*, this heuristic looks one move ahead, calculating the difference between player and opponent legal moves length, using this value as the score of the current game state.

3. custom_score_3: the idea for this heuristic is to assign a better score for the moves that belong to the centre of the board, penalising the moves that bring the player close to the edge.

Performances

To evaluate these three heuristics, I ran the *tournament.py* program, simulating 10 tournaments running 20 matches against each opponent.

I can observe that the AB_Custom2 agent, that implement the custom_score_2 heuristic, performs better than all the other heuristics considered. The gain in performance is around 2%. Table 1 and Table 2 show the results for all the agents applying different heuristics in 10 tournaments, each consisting of 20 matches per opponent.

	Tournament N.	AB_Improved	AB_Custom	AB_Custom2	AB_Custom3
0	1	58.6	57.1	68.6	60.0
1	2	64.3	62.9	68.6	61.4
2	3	63.9	64.6	65.8	60.7
3	4	61.4	57.1	62.9	62.1
4	5	63.6	62.9	62.1	62.9
5	6	63.4	60.7	64.3	56.4
6	7	67.1	62.9	69.3	66.4
7	8	67.3	61.2	61.2	60.2
8	9	63.3	57.2	60.2	53.1
9	10	58.6	62.9	67.1	60.0

Table 1 - Results for all the agents

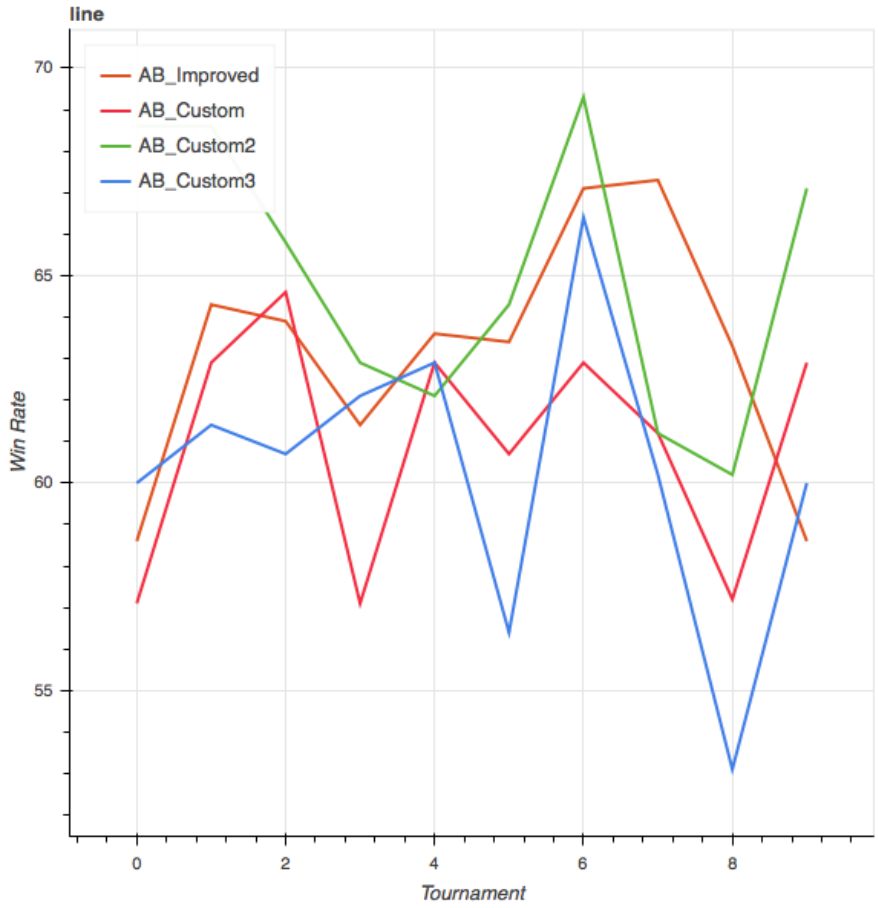

Figure 1 - Results for all the agents

	AB_Improved	AB_Custom	AB_Custom2	AB_Custom3
count	10.000000	10.000000	10.000000	10.000000
mean	63.150000	60.950000	65.010000	60.320000
std	2.971438	2.835587	3.337481	3.587261
min	58.600000	57.100000	60.200000	53.100000
25%	61.875000	58.075000	62.300000	60.000000
50%	63.500000	62.050000	65.050000	60.450000
75%	64.200000	62.900000	68.225000	61.925000
max	67.300000	64.600000	69.300000	66.400000

Table 2 – Tournaments descriptive statistics

The *custom_score_2* heuristic evaluation function is recommended because:

- It is based on the number of moves looking ahead one move in the future. Due to L-shaped knight like moves, it's difficult to predict the value of a game state by just counting the number of immediately available moves like in the Improved heuristic provided. Hence one move look-ahead should provide a better heuristic function.

- Most of times *custom_score_2* heuristic functions beat all other evaluation functions in head to head competitions.

Match #	Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3	
		Won	Lost	Won	Lost	Won	Lost	Won	Lost
1	Random	35	5	32	8	31	9	32	8
2	MM_Open	24	16	26	14	27	13	24	16
3	MM_Center	31	9	30	10	30	10	31	9
4	MM_Improved	23	17	26	14	26	14	21	19
5	AB_Open	19	21	24	16	19	21	22	18
6	AB_Center	25	15	22	18	21	19	23	17
7	AB_Improved	22	18	21	19	23	17	17	23

Match #	Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3	
		Won	Lost	Won	Lost	Won	Lost	Won	Lost
1	Random	32	8	31	9	37	3	33	7
2	MM_Open	31	9	25	15	28	12	28	12
3	MM_Center	31	9	28	12	37	3	30	10
4	MM_Improved	24	16	24	16	28	12	29	11
5	AB_Open	20	20	22	18	16	24	22	18
6	AB_Center	28	12	24	16	25	15	24	16
7	AB_Improved	22	18	22	18	23	17	20	20

- As noted from the results in Table 1 and Table 2, the *custom_score_2* heuristic perform better, and can be recommended.