AIND Isolation game playing agent report

Andrzej Wytyczak-Partyka

December 5, 2017

The task was to implement an agent playing the Isolation game, using Minimax with alpha-beta pruning and iterative deepening. The agent would use one of 3 heuristics for assesing current board state during termination of each search. Termination happens either when end-game is reached or when the search reaches specified depth.

The heuristic implemented were:

- AB_Custom - For first 5 moves - use the distance from center heuristic, Later in the game - use the number of my moves - number of opponent moves
- AB_Custom_2 - For first 40 moves - use distance between opponent's pieces, later in the game - use the square of number of available moves
- AB_Custom_3 - For first 3 moves - assign High score for center move, later assign higher points to locations near the center, finally use square of number of available moves

I examined the results of 20 runs of the tournament.py script (shown in Figure 3) and ran simple statistical analysis, as shown in Figure 1.

The result of the analysis (as seen in Figure 2) confirms that AB_Custom_3 is far worse than AB_Improved, as well as that AB_Custom and AB_Custom_2 are not statistically different from AB_Improved.

```
print("h0 is that AB_Improved has the same performance as AB_Custom")

[stat,pvalue] = stats.ttest_ind(df['AB_Improved'], df['AB_Custom'], 0)
if(pvalue < 0.05):
    print('h0 rejected - AB_Improved does not have the same performance as AB_Custom')
else:
    print('h0 accepted - AB_Improved does have the same performance as AB_Custom')

print("h0 is that AB_Improved has the same performance as AB_Custom_2")

[stat,pvalue] = stats.ttest_ind(df['AB_Improved'], df['AB_Custom_2'], 0)
if(pvalue < 0.05):
    print('h0 rejected - AB_Improved does not have the same performance as AB_Custom_2')
else:
    print('h0 accepted - AB_Improved does have the same performance as AB_Custom_2')

print("h0 is that AB_Improved has the same performance as AB_Custom_3")

[stat,pvalue] = stats.ttest_ind(df['AB_Improved'], df['AB_Custom_3'], 0)
if(pvalue < 0.05):
    print('h0 rejected - AB_Improved does not have the same performance as AB_Custom_3')
else:
    print('h0 accepted - AB_Improved does have the same performance as AB_Custom_3')
```

Figure 1: Python code to determine if heuristics differ from AB_Improved

```
h0 is that AB_Improved has the same performance as AB_Custom
h0 accepted - AB_Improved does have the same performance as AB_Custom
h0 is that AB_Improved has the same performance as AB_Custom_2
h0 accepted - AB_Improved does have the same performance as AB_Custom_2
h0 is that AB_Improved has the same performance as AB_Custom_3
h0 rejected - AB_Improved does not have the same performance as AB_Custom_3
```

Figure 2: Results of t-test verification

#	AB_Improved	AB_Custom	AB_Custom_2	AB_Custom_3
0	52.9	60.0	61.4	60.0
1	61.4	57.1	67.1	55.7
2	67.1	62.9	70.0	55.7
3	61.4	65.7	64.3	50.0
4	61.4	54.3	60.0	57.1
5	68.6	64.3	64.3	64.3
6	70.0	65.7	74.3	62.9
7	65.7	62.9	61.4	64.3
8	67.1	60.0	64.3	55.7
9	67.1	57.1	54.3	58.6
10	67.1	57.1	62.9	58.6
11	62.9	62.9	55.7	55.7
12	64.3	67.1	58.6	52.9
13	71.4	61.4	58.6	60.0
14	67.1	61.4	67.1	64.3
15	60.0	61.4	58.6	61.4
16	54.3	58.6	60.0	60.0
17	61.4	65.7	64.3	55.7
18	64.3	68.6	70.0	65.7
19	61.4	60.0	70.0	64.3

Figure 3: Percent of games won during 20 runs of ‘python tournament.py‘

	AB_Improved	AB_Custom	AB_Custom_2	AB_Custom_3
count	20.000000	20.000000	20.000000	20.000000
mean	63.845000	61.710000	63.360000	59.145000
std	4.762073	3.788056	5.244887	4.337047
min	52.900000	54.300000	54.300000	50.000000
25%	61.400000	59.650000	59.650000	55.700000
50%	64.300000	61.400000	63.600000	59.300000
75%	67.100000	64.650000	67.100000	63.250000
max	71.400000	68.600000	74.300000	65.700000

Figure 4: Descriptive statistics for 20 runs of tournament

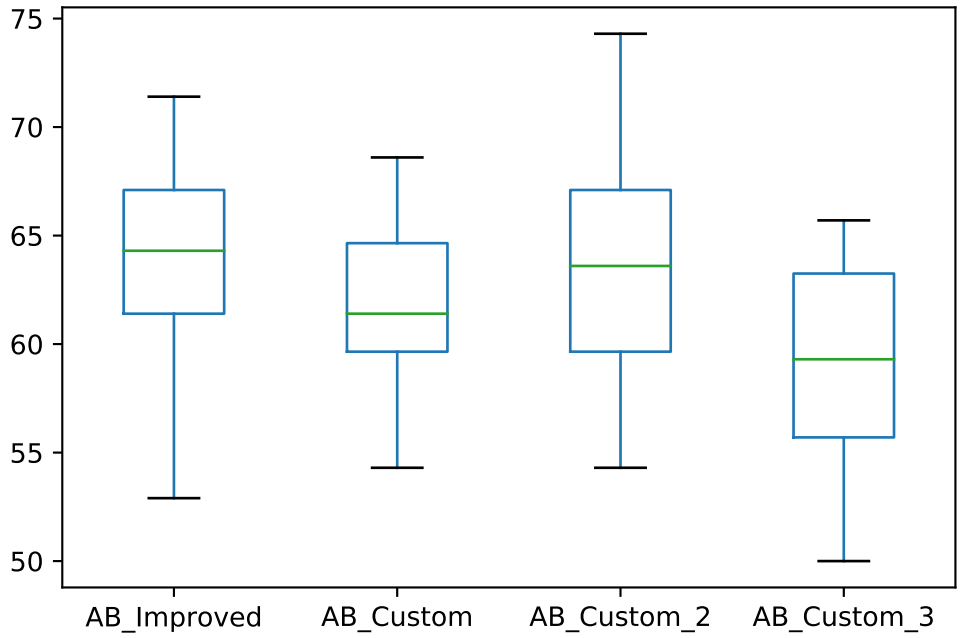

Figure 5: Box plot for AB_Improved and custom heuristics