Choosing a heuristic

How to compare heuristics

To choose the best heuristic, we must define what metric to use to compare the performance of a new evaluation function against the improved scored evaluation function is. Because we do not know the strategy employed by the adversary, it is not wise to simply play them against each other and choose the one that can defeat the Improved ID most times. It is plausible to have a scenario where an evaluation function reliably beat ID_Improved but losses against other strategies over 50% of the time.

I decided to run five tournaments, log the results, and compare the percentage of win games using a T-Student test with the “greater” alternative. Choosing the function for which the data suggest the mean winning percentage is higher than the winning percentage of the agents running with the improved score evaluation function.

Using a T-Test instead of a Z-Test is a way to avoid having to run a higher number of tournaments, which turns out to be time-consuming on my laptop. I set the significance level at 95% and the alternative hypothesis to be that the custom function had a greater mean number of victories.

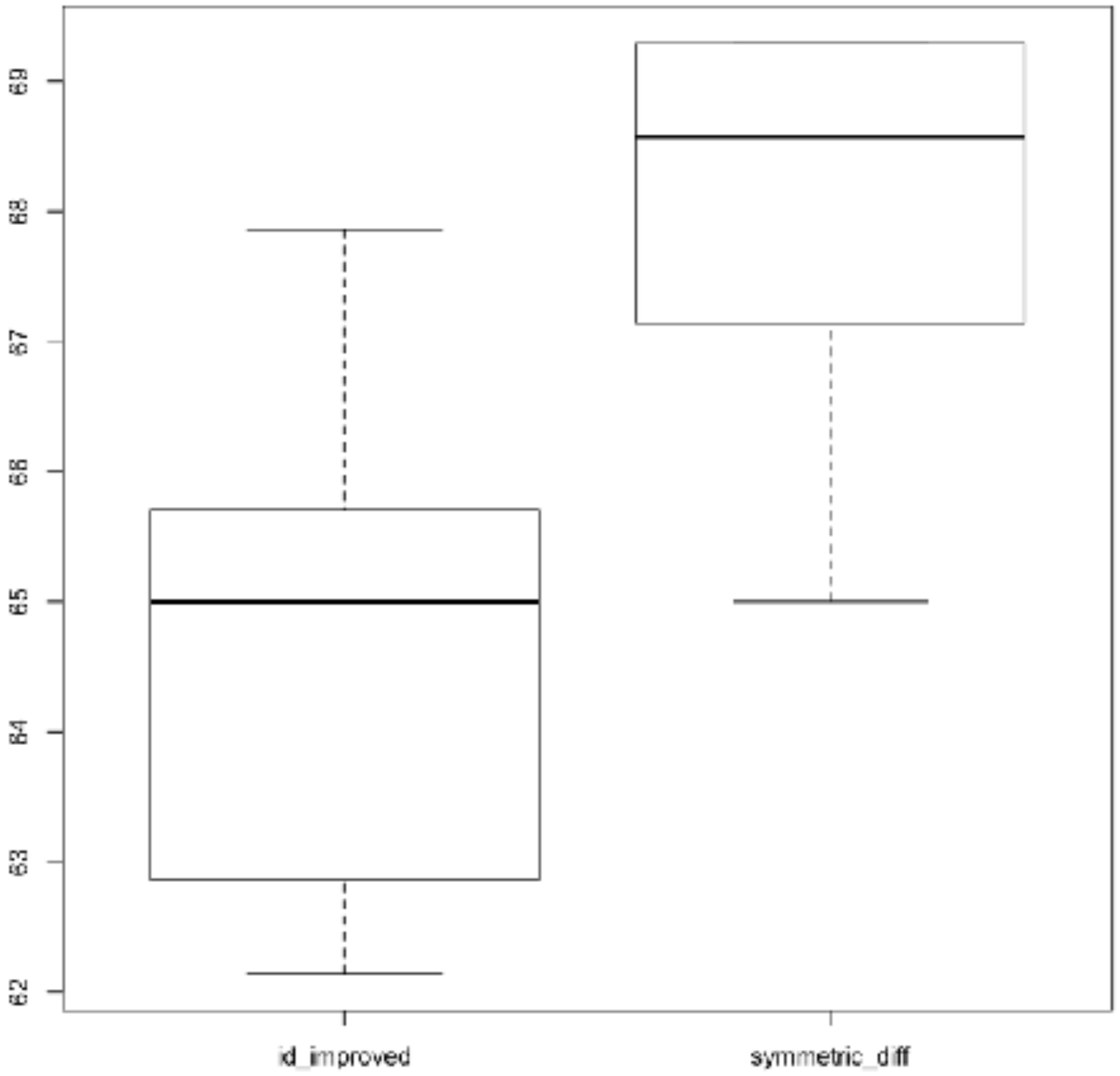
The null hypothesis is that the custom function performs equal or worst to the ID_Improved function.

The 3 functions I am presenting for review are: difference_of_symmetric_set_difference, fractional_3_deep and fractional_2_deep

Function 1 - Difference of Symmetric Differences Best Heuristic

This function calculates the difference of the symmetric difference of moves of the players; I estimate the number of moves by looking two steps ahead. To avoid having to incur the cost of constructing the boards, I approximate the value by using the moves from the function instead of forecast_move.

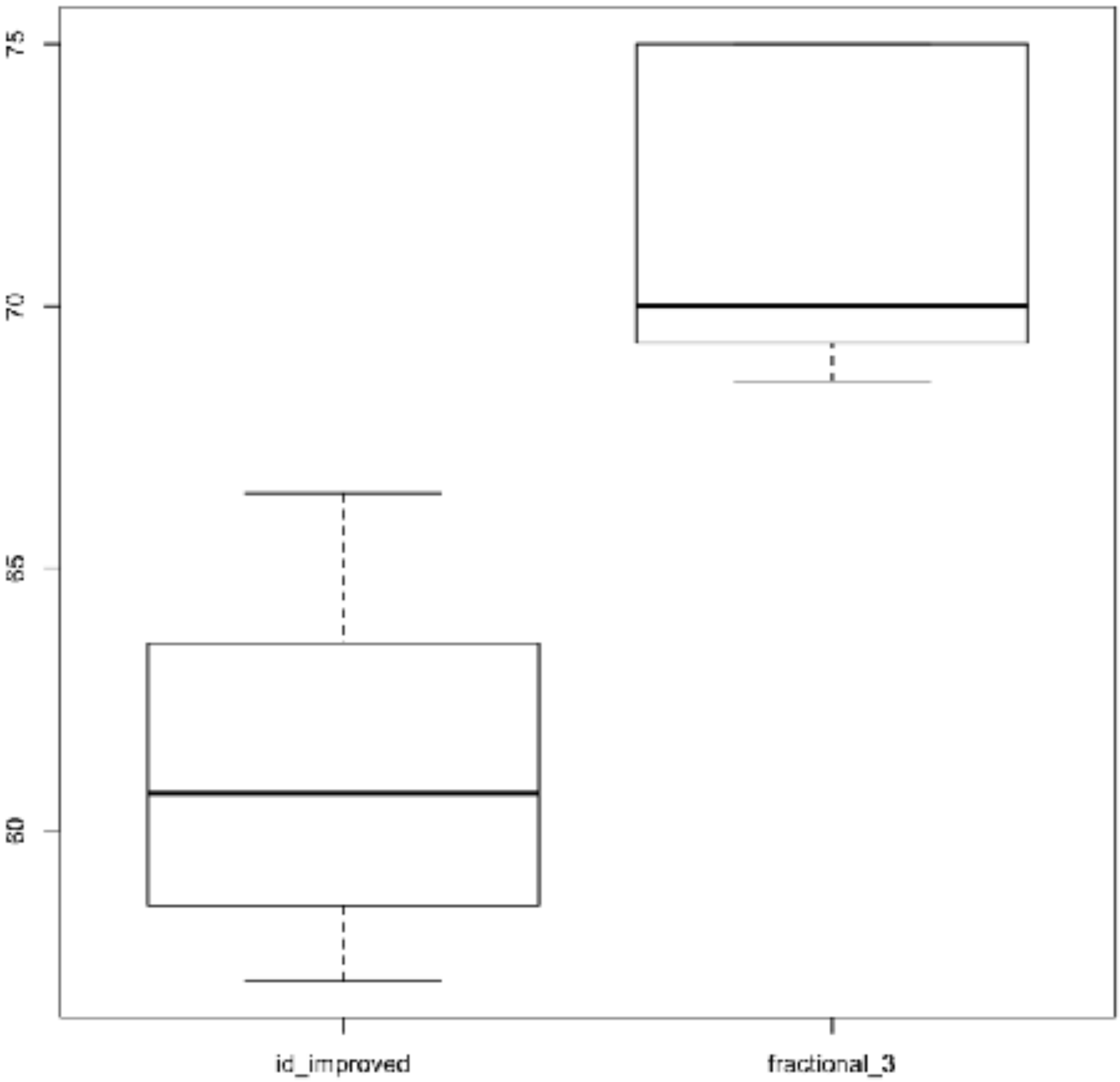
The comparison of performance allows me to reject the null hypothesis, with a p-value of 0.02235.



Function 2 - Fraction of moves vs. opponent moves with 2 step look ahead

This function calculates the ratio of the moves vs. opponent moves, looking to steps ahead. I used the same strategy as with the previous heuristic to avoid the cost of calling forecast_move.

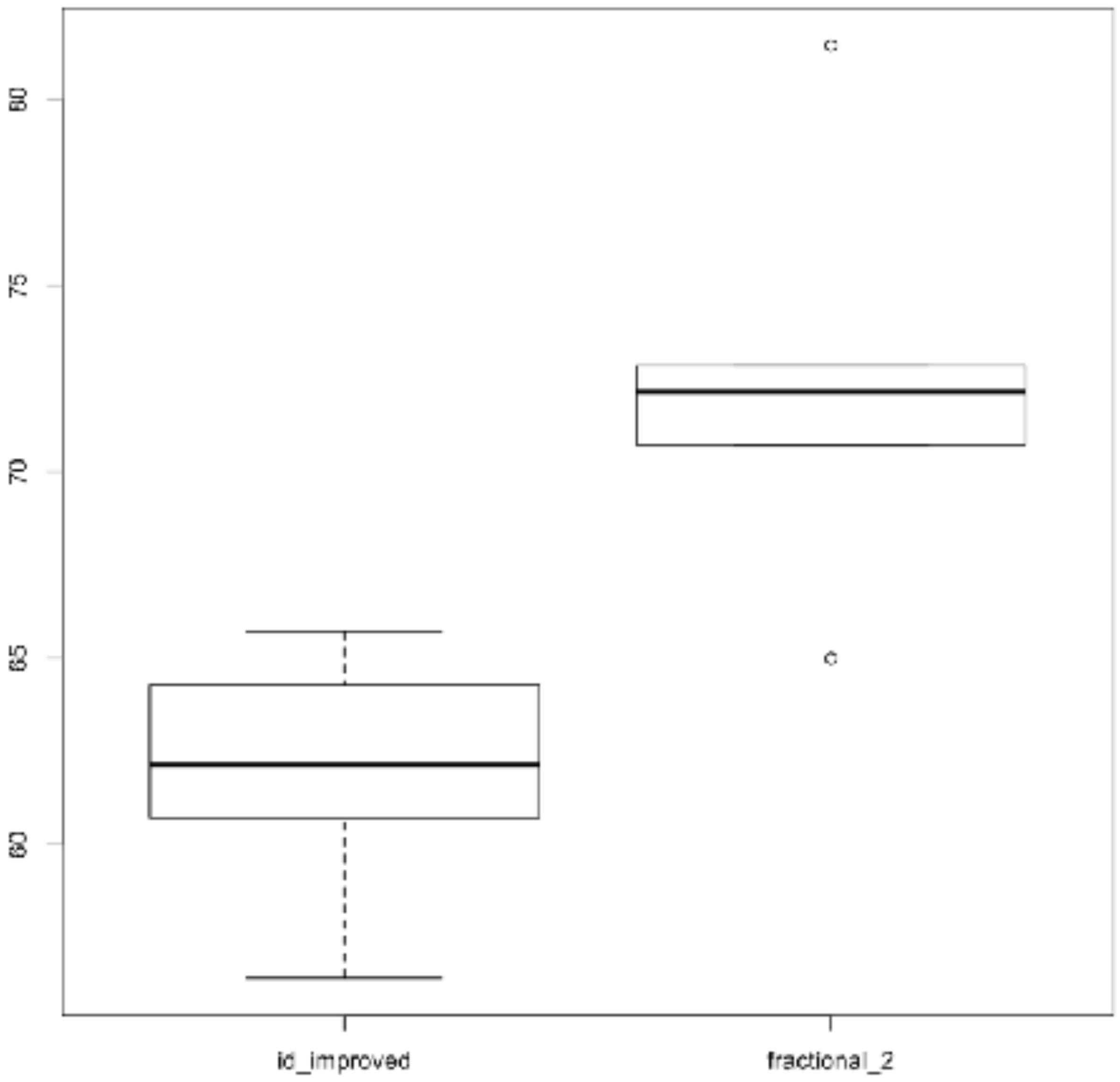
The comparison of performance allows me to reject the null hypothesis, with a p-value of 0.0008564.



Function 3 - Fraction of moves vs. opponent moves with 1 step look ahead

The idea of this function is the same as before but looking only one move ahead instead of two.

The comparison of performance allows me to reject the null hypothesis, with a p-value of 0.006086.



Choosing the Heuristic

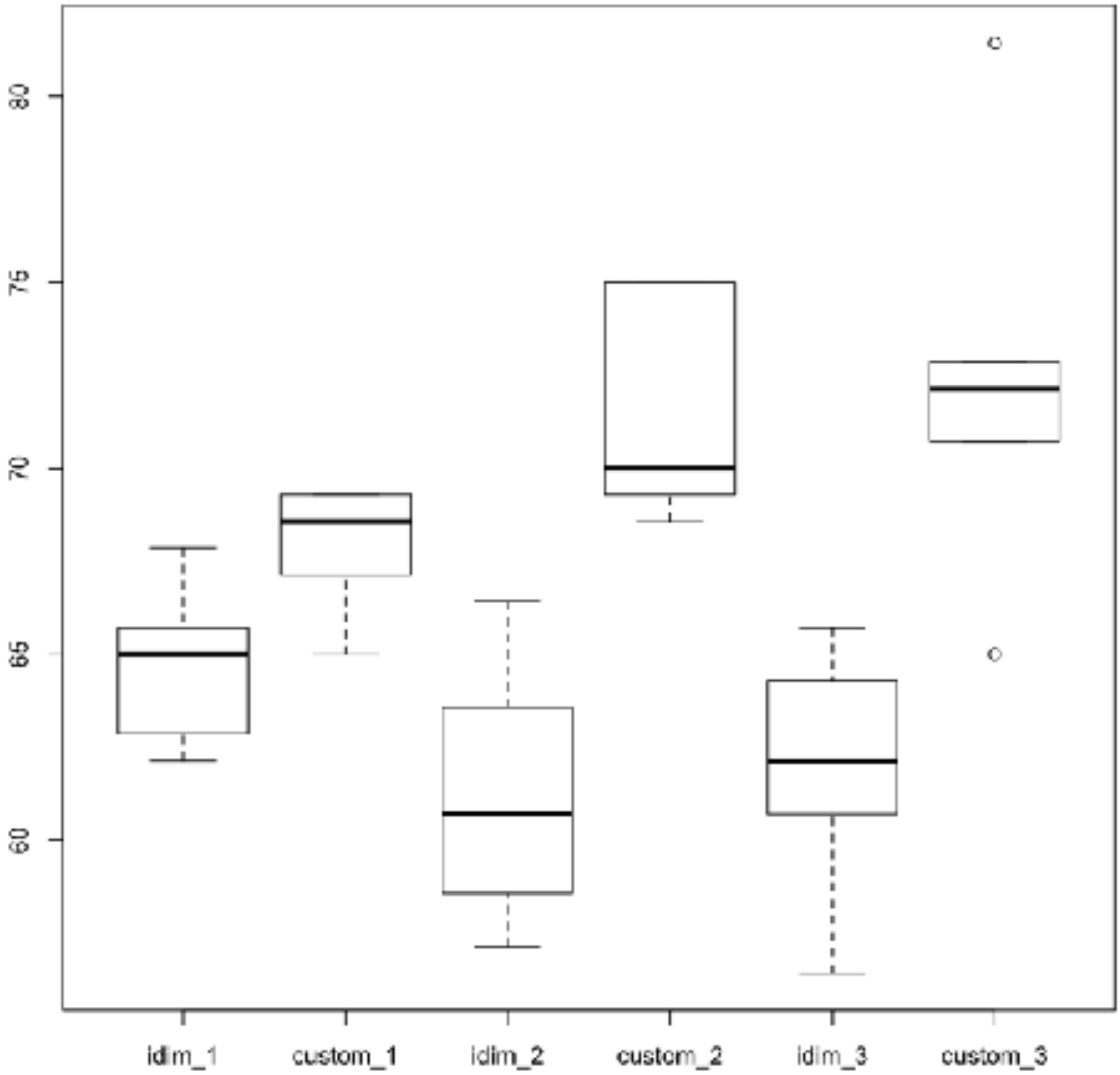
The three proposed functions perform better than ID_Improved based on the statistical test. I was expecting some of the functions to perform the same or worst than ID improved because the extra cost of calculating the function would mean that the search had to be shallower. I think this was not the case because a combination of the following reasons:

- Knight isolation has a significant smaller branching factor than Queen Isolation, a lower branching factor means that there is less penalty associated with a more complex heuristic function
- I added a little optimisation on the expansion step, instead of visiting the states in the order they are generated I visited them based on the number of moves available to the active player. Visiting the ones with the most possible moves earlier can result in more pruning, and consequently less penalty for a complicated heuristic.
- I did not use forecast_move inside the heuristic function and avoided the expensive step of creating new boards during the calculation.

Chosen Heuristic

To propose a heuristic I decided to use the same tool used to compare the individual functions with ID improved. Starting with a General box plot to choose the two apparent best-performing heuristics and analysing them using a t-test.

The best performing heuristics are the fractional ones, a direct comparison between the two does not allow me to reject the null hypothesis (p-value=0.3923). In this case, then I suggest using the heuristic that is less complicated to calculate which is fractional with two deep look-ahead which is labeled custom_3 in the box chart below.


80
75
70
65
60
idim_1
custom_1
idim_2
custom_2
idim_3
custom_3