# Analysis of heuristic functions in Isolation game

Totally 7 heuristic functions are tested. Listed below:

|  |  |
| --- | --- |
| Function name | Function implementation |
| heuristic\_diff\_score | own\_moves - opp\_moves |
| heuristic\_score\_var\_weight | own\_moves/game.move\_count |
| heuristic\_diff\_score\_with\_count | own\_moves - opp\_moves - game.move\_count |
| heuristic\_oppononet\_negtive\_score | -opp\_moves |
| heuristic\_own\_score\_fixed\_weight\_1 | 2\*own\_moves - opp\_moves |
| heuristic\_own\_score\_fixed\_weight\_2 | own\_moves/2 - opp\_moves |
| heuristic\_oppo\_score\_fixed\_weight | own\_moves - 2\*opp\_moves |

And the wining rate data of different heutistic functions listed below:

|  |  |  |  |
| --- | --- | --- | --- |
| Function name | Function wining rate | Reference wining rate | delta |
| heuristic\_diff\_score | 66.43% | 67.86% | -1.43% |
| heuristic\_score\_var\_weight | 71.43% | 72.86% | -1.43% |
| heuristic\_diff\_score\_with\_count | 64.29% | 70.71% | -6.42% |
| heuristic\_oppononet\_negtive\_score | 65.00% | 69.29% | -3.71 |
| heuristic\_own\_score\_fixed\_weight\_1 | 64.29% | 68.57% | -4.26% |
| heuristic\_own\_score\_fixed\_weight\_2 | 75.71% | 67.86% | 7.05% |
| heuristic\_oppo\_score\_fixed\_weight | 70.71% | 66.43% | 3.28% |

The first function is actually the same with the ID\_improved function, but we can see the wining rate is somehow different. This means the wining rate is not stable even for the same heuristic in the same tournament. So if your function cannot over-perform the reference more than like 2 percent delta constantly, it is not a valid prof that your function is better.

heuristic\_score\_var\_weight and heuristic\_diff\_score\_with\_count introducing the move\_count variable which doesn’t help and it improves the calculation time, affecting the search depth. heuristic\_oppononet\_negtive\_score simply uses the negative score of the opponent’s move but its results is better than expected. I think the reason is when the heuristic function takes place, it is actually the opponents turn to move, so my\_move is actually not a correct number (opponent’s move is the correct number) so will introduce some error. Moving on, let’s improve the weight of my\_move using heuristic\_own\_score\_fixed\_weight\_1, the performance becomes worse. Then moving on to lower my\_move’s weight using heuristic\_own\_score\_fixed\_weight\_2， the performance finally out-performs the ID\_improved. With these 3 functions, we can see that know yourself and know your opponent’s better(which means take opponent’s with more weight) is a good strategy. Let’s use another heuristic heuristic\_oppo\_score\_fixed\_weight to improve the weight of the opponent’s moves , it also out performs the ID\_improved, which validates our point.

Since heuristic\_own\_score\_fixed\_weight\_2 which lower’s my\_move’s weight to 0.5 has the best performance, so I choose this function in the task. From other unsatisfactory heuristics, we can learn that 1. It is not easy to out perform the ID\_improved heuristic. 2. Sometimes more variables and thus more calculation is not worthy it, because it will case the search depth shallow. 3. Know yourself and know your enemy better applies to this problem, sometimes we can apply some life wisdom to solving AI issues.