

Heuristic Analysis

The custom heuristic functions which are taken into consideration for this isolation game are the following:

1. $eval_param_score_fn(game, player, x, y, z)$
2. $eval_normalize_score_fn(game, player)$
3. $eval_reduce_opp_score_fn(game, player)$

1. $eval_param_score_fn(game, player, x, y, z)$:

Parametrized evaluation function. Here x,y,z are tuning parameter. Evaluation score is derived from following equation.

The matches are played with different variable x,y, z

Function fn x

- 1 $fn1(x=1, y=0.5, z=0)$
2. $fn2(x=1, y=0.5, z=1)$
3. $fn3(x=1, y=0.25, z=1)$
4. $fn4(x=0.5, y=1, z=1)$
5. $fn5(x=0.75, y=0.25, z=1)$
6. $fn6(x=0.5, y=0.75, z=2)$
7. $fn7(x=1, y=0.5, z=0.5)$

Match chart:

Notation - fn x - i where i = match iteration .

Example: $fn2-3$ indicates $fn2$ evaluation function is used and it's 3rd

the iteration

Table 1 - Evaluation of ID_Improved.

ID_IMPROVED		Win Count of 20											
S	Opponent	Fn1-1	Fn2-1	Fn3-1	Fn4-1	Fn5-1	Fn6-1	Fn7-1	Fn2-2	Fn2-3	Fn2-4	Fn2-5	Fn2-Mean
1	Random	20	18	19	18	18	17	19	19	20	17	19	19
2	MM_Null	18	16	19	18	18	15	9	16	13	17	18	16
3	MM_Open	12	11	13	14	15	13	17	15	13	16	15	14
4	MM_Improved	14	13	14	10	9	14	11	10	14	12	13	12
5	AB_Null	15	16	16	15	14	9	13	16	16	15	16	16
6	AB_Open	8	9	13	12	11	10	13	15	14	16	13	13
7	AB_Improved	14	14	10	13	11	11	12	11	13	12	12	12
Winning %		69.29	69.29	74.29	74.29	68.57	63.57	67.14	72.86	73.57	75	75.71	72.86

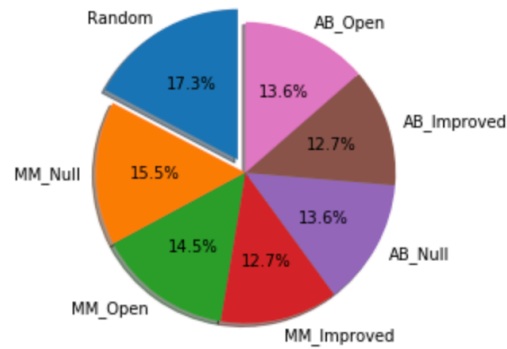
Table 2 - Evaluation of Student

STUDENT		Win Count of 20											
S	Opponent	Fn1-1	Fn2-1	Fn3-1	Fn4-1	Fn5-1	Fn6-1	Fn7-1	Fn2-2	Fn2-3	Fn2-4	Fn2-5	Fn2-Mean
1	Random	16	20	18	17	18	17	19	18	20	17	18	19
2	MM_Null	18	18	12	15	15	20	16	16	18	17	16	17
3	MM_Open	12	16	13	15	14	15	13	15	18	16	14	16
4	MM_Improved	15	11	14	12	15	10	16	15	11	17	15	14
5	AB_Null	16	15	13	13	10	16	14	15	11	18	16	15
6	AB_Open	14	15	13	15	15	11	14	15	15	13	15	15
7	AB_Improved	18	15	13	12	15	14	12	12	14	13	14	14
Winning %		77.86	78.57	68.57	70.71	72.86	73.57	74.28	7.14	80	79.29	77.14	78.57

Based on Table 2 data the relative performance of fn_x is shown below

After multiple tries, fn2 shows better winning ratio. Fn2 is iterated multiple time to calculate the fn2-Mean. Mean winning percentage of fn2= 78.57%

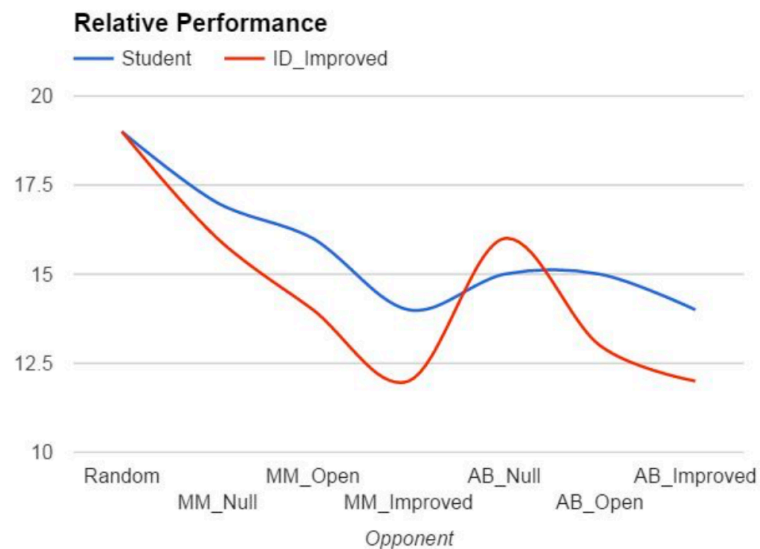
Fn Mean



Bench match comparison:

Student vs ID_Improved Mean winning % of D_Improved : **72.86%**

Mean winning % of Student : **78.57%**



2. ***eval_normalize_score_fn(game, player)*** : The function evaluates based on the number of occupied space. When board positions are less than half occupied, agent weighs to reduce opponent's legal move. (Aggressive play)

Match chart:

Notation - fn- i where i = match iteration

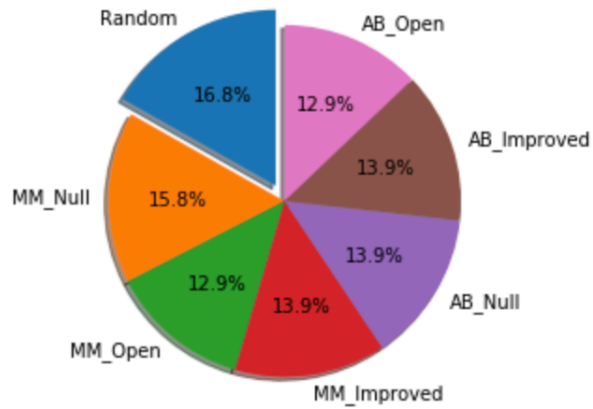
Table 1 - Evaluation of ID_Improved.

IMPROVED		Win Count of 20							
SN	Opponent	fn-1	Fn-2	Fn-3	Fn-4	Fn-5	Fn-6	Fn-7	fn-Mean
1	Random	18	17	19	15	18	11	19	17
2	MM_Null	17	15	19	11	17	19	18	17
3	MM_Open	16	11	15	10	15	10	8	12
4	MM_Improved	14	12	14	11	16	11	13	13
5	AB_Null	15	16	13	15	18	18	13	15
6	AB_Open	15	14	14	11	18	12	9	13
7	AB_Improved	13	12	10	15	13	15	12	13
	Winning %	77.14	69.29	74.29	62.86	77.86	68.57	64.29	71.42

Table 2 - Evaluation of Student

Student		Win Count of 20							
SN	Opponent	fn-1	Fn-2	Fn-3	Fn-4	Fn-5	Fn-6	Fn-7	fn-Mean
1	Random	18	15	17	19	18	14	17	17
2	MM_Null	18	15	17	16	18	13	18	16
3	MM_Open	12	12	15	12	15	14	13	13
4	MM_Improved	14	12	15	15	15	13	13	14
5	AB_Null	18	12	15	12	17	13	12	14
6	AB_Open	13	13	12	14	12	15	12	13
7	AB_Improved	12	13	17	14	14	15	11	14
	Winning %	75	65.71	77.14	72.87	77.86	69.29	68.57	72.14

Fn Mean

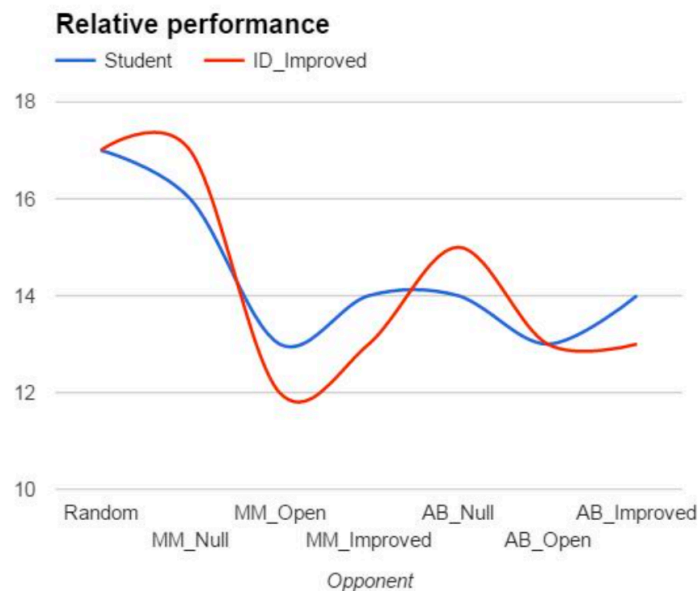


Bench match

comparison:

Student vs ID_Improved Mean winning % of D_Improved **71.42%**

Mean winning % of Student : **72.14%**



3. *eval_reduce_opp_score_fn(game, player)*: This evaluation function solely based on opponent's number of legal moves.

Opp_move = Move count available for opponent player

Match chart:

Notation - fn- i where i = match iteration

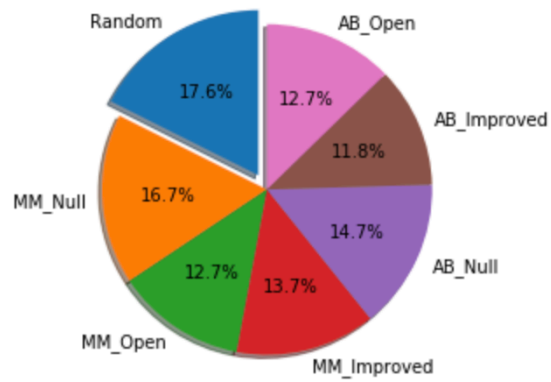
Table 1 - Evaluation of ID_Improved.

IMPROVED		Win Count of 20							
SN	Opponent	fn-1	Fn-2	Fn-3	Fn-4	Fn-5	Fn-6	Fn-7	fn-Mean
1	Random	19	15	19	20	17	19	18	18
2	MM_Null	16	15	18	19	15	20	17	17
3	MM_Open	12	12	13	14	15	14	14	13
4	MM_Improved	11	16	11	14	12	13	13	13
5	AB_Null	15	14	12	16	13	16	15	14
6	AB_Open	15	12	13	8	13	10	15	12
7	AB_Improved	14	13	14	12	12	14	14	13
	Winning %	72.86	69.28	71.43	69.28	75.71	75.71	75.71	71.42

Table 2 - Evaluation of Student

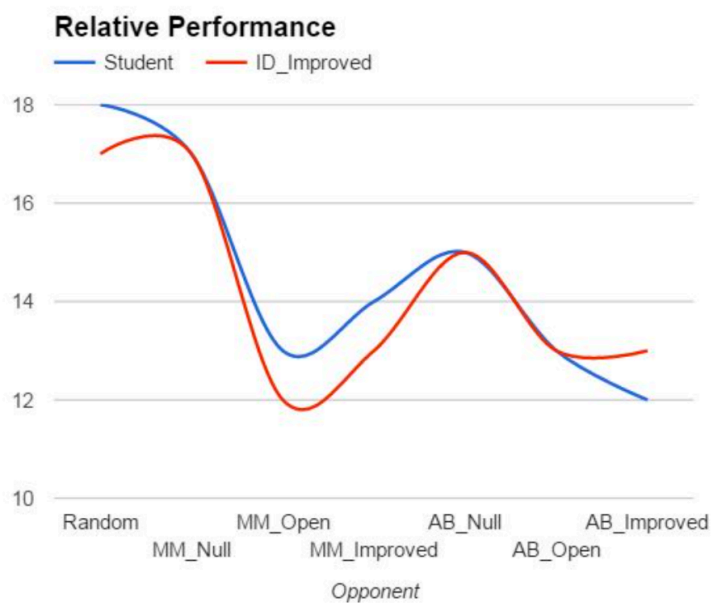
Student		Win Count of 20							
SN	Opponent	fn-1	Fn-2	Fn-3	Fn-4	Fn-5	Fn-6	Fn-7	fn-Mean
1	Random	18	19	19	19	17	19	18	18
2	MM_Null	15	17	17	18	16	17	17	17
3	MM_Open	11	14	12	12	14	11	16	13
4	MM_Improved	13	12	18	14	14	12	12	14
5	AB_Null	12	13	15	15	18	16	18	15
6	AB_Open	13	12	15	13	14	12	14	13
7	AB_Improved	15	12	10	13	12	13	12	12
	Winning %	69.28	70.71	75.71	74.29	75	70.71	77.14	72.86

Fn Mean



Bench match comparison:

Student vs ID_Improved Mean winning % of D_Improved **71.42%**
 Mean winning % of Student : **72.86 %**



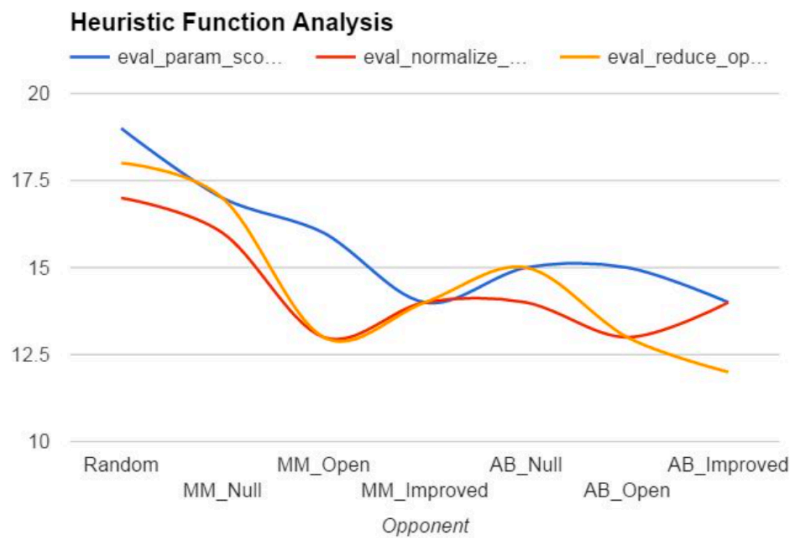
Conclusion:

Relative comparison of heuristic function result:

Mean winning % using eval_param_score_fn : 78.57%

Mean winning % using eval_normalize_score_fn Mean winning % using : 72.14%

eval_reduce_opponent_score_fn : 72.86



Therefore on above analysis results, **eval_param_score_fn** is the better choice compared to other two. It slightly performs better against all opponents. By tweaking the parameter, performance can be further improved and by learning using knowledge base of game results, the parameter can be optimized.