# Heuristic Analysis

## Introduction

The analysed position evaluation functions are all a variation of counting the number of legal moves minus the number of legal moves available to the opponent:

Count(Moves) – Count(OpponentMoves)

Realising that some moves are more valuable than others one can sum the value of each move instead of just counting the moves. Subsequently the evaluation function was defined as:

Sum(Map(n->Pow(Value(n),N)),Moves))

– Weight \* Sum(Map(n->Pow(Value(n),N)), OpponentMoves))

whereas:

|  |  |
| --- | --- |
| Value | Defines the value of single move |
| Pow | Raises the value to the power of N to give more weight to valuable moves |
| Weight | Is an additional factor that helps eliminating especially valueable oppenent’s moves |

### Value of Move

The value of a move is defined as the number of moves available on an empty board from a certain position. These numbers on a 7x7 board are:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Row/Column | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 0 | 2 | 3 | 4 | 4 | 4 | 3 | 2 |
| 1 | 3 | 4 | 6 | 6 | 4 | 4 | 3 |
| 2 | 4 | 6 | 8 | 8 | 8 | 6 | 4 |
| 3 | 4 | 6 | 8 | 8 | 8 | 6 | 4 |
| 4 | 4 | 6 | 8 | 8 | 8 | 6 | 4 |
| 5 | 3 | 4 | 6 | 6 | 4 | 4 | 3 |
| 6 | 2 | 3 | 4 | 4 | 4 | 3 | 2 |

These numbers were then normalized to values as shown below:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Row/Column | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 0 | 0.25 | 0.38 | 0.5 | 0.5 | 0.5 | 0.38 | 0.25 |
| 1 | 0.38 | 0.5 | 0.75 | 0.75 | 0.5 | 0.5 | 0.38 |
| 2 | 0.5 | 0.75 | 1 | 1 | 1 | 0.75 | 0.5 |
| 3 | 0.5 | 0.75 | 1 | 1 | 1 | 0.75 | 0.5 |
| 4 | 0.5 | 0.75 | 1 | 1 | 1 | 0.75 | 0.5 |
| 5 | 0.38 | 0.5 | 0.75 | 0.75 | 0.5 | 0.5 | 0.38 |
| 6 | 0.25 | 0.38 | 0.5 | 0.5 | 0.5 | 0.38 | 0.25 |

## Variations

To test the idea following instances of the evaluation functions were defined an analysed:

|  |  |  |  |
| --- | --- | --- | --- |
| **Functionnames** | N = 1 | N = 2 (squared) | N = 3(cubed) |
| Weight = 1 | **ValueScore** | **Value2Score** | **Value3Score** |
| Weight = 2 | **ValueWeightedScore** | **Value2WeightedScore** | **Value3WeightedScore** |

For example, function Value2WeightedScoresquares all Values and applies a weight of 2.

## Analysis

The above defined evaluation functions were tested using the Script *tournament.py.* The functions were then compared to the function AB\_Improved which served as a benchmark. 10 rounds were played using the functions ValueScore, Value2Score and Value3Score. Another 10 rounds were played using the functions ValueWeightedScore, Value2WeightedScore and Value3WeightedScore. The “Win Rate” of a certain function was divided by the “Win Rate” of the benchmark function AB\_Improved. Following table shows the results:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Algorithm / Tournament A** | Rnd 1 | Rnd 2 | Rnd 3 | Rnd 4 | Rnd 5 | Rnd 6 | Rnd 7 | Rnd 8 | Rnd 9 | Rnd 10 |
| AB\_Improved | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| ValueScore | 1.13 | 1.02 | 0.94 | 0.91 | 1.22 | 0.94 | 0.96 | 1.02 | 0.96 | 0.98 |
| Value2Score | 1.07 | 1.12 | 1.02 | 0.96 | 1.11 | 1.00 | 1.09 | 1.00 | 1.12 | 0.92 |
| Value3Score | 1.09 | 0.96 | 1.06 | 0.92 | 1.16 | 1.00 | 1.06 | 1.02 | 0.96 | 0.87 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Algorithm / Tournament B** | Rnd 1 | Rnd 2 | Rnd 3 | Rnd 4 | Rnd 5 | Rnd 6 | Rnd 7 | Rnd 8 | Rnd 9 | Rnd 10 |
| AB\_Improved | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| ValueWeightedScore | 1.00 | 1.14 | 1.21 | 0.94 | 1.03 | 1.00 | 0.90 | 1.06 | 0.90 | 1.02 |
| Value2WeightedScore | 1.02 | 1.06 | 1.28 | 1.08 | 1.00 | 1.04 | 1.00 | 1.04 | 1.04 | 0.96 |
| Value3WeightedScore | 0.96 | 0.98 | 1.19 | 0.92 | 1.06 | 1.04 | 0.98 | 1.00 | 1.08 | 0.98 |
|  |  |  |  |  |  |  |  |  |  |  |
|  | Best in Round | | |  |  |  |  |  |  |  |
|  | Worst in Round | | |  |  |  |  |  |  |  |

Following table summarises the “Win Rates” and algorithms chosen:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ranking** | **Algorithm / Tournament A** | Nr Best | Nr Worst | Nr Best/Nr Worst | Nr > 1 |
|  | AB\_Improved | 2 | 2 | 1 | 0 |
|  | ValueScore | 2 | 5 | 0.4 | 4 |
| 2.) | Value2Score | 3 | 0 | - | 6 |
|  | Value3Score | 1 | 2 | 0.5 | 5 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  | **Algorithm / Tournament B** |  |  |  |  |
|  | AB\_Improved | 0 | 2 | 0 | 0 |
| 3.) | ValueWeightedScore | 3 | 3 | 1 | 5 |
| 1.) | Value2WeightedScore | 4 | 1 | 4 | 7 |
|  | Value3WeightedScore | 3 | 3 | 1 | 4 |

Best function was “Value2WeightedScore” which was 4 out of 10 rounds the best algorithm and only once the worst. It was 7 out of 8 times better than AB\_Improved.