# 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 (**VS**) | Value2Score (**V2S**) | Value3Score (**V3S**) |
| Weight = 2 | ValueWeightedScore (**VWS**) | Value2WeightedScore  (**V2SW**) | Value3WeightedScore  (**V3WS**) |

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*. Each evaluation function was tested in a player (agent) that implements min/max-search with alpha/beta pruning. Those players played against a set of reference player. Each pair of player played 50 matches (NUM\_MATCHES=50). Following is the output of one tournament:

Match # Opponent AB\_Improved **VS V2S V3S VWS V2WS V3WS**

Won | Lost Won | Lost Won | Lost Won | Lost Won | Lost Won | Lost Won | Lost

1 Random 96 | 4 94 | 6 94 | 6 98 | 2 92 | 8 91 | 9 95 | 5

2 MM\_Open 80 | 20 81 | 19 82 | 18 79 | 21 77 | 23 76 | 24 81 | 19

3 MM\_Center 89 | 11 90 | 10 84 | 16 90 | 10 88 | 12 94 | 6 92 | 8

4 MM\_Improved 63 | 37 79 | 21 72 | 28 73 | 27 75 | 25 76 | 24 76 | 24

5 AB\_Open 50 | 50 64 | 36 55 | 45 54 | 46 54 | 46 53 | 47 51 | 49

6 AB\_Center 57 | 43 60 | 40 62 | 38 61 | 39 57 | 43 66 | 34 62 | 38

7 AB\_Improved 50 | 50 56 | 44 52 | 48 51 | 49 55 | 45 59 | 41 46 | 54

----------------------------------------------------------------------------------------------------------------

**Win Rate: 69.3% 74.9% 71.6% 72.3% 71.1% 73.6% 71.9%**

The Win Rate shows that all evaluation functions performed better than AB\_Improved which is a min/max-search with alpha/beta pruning using “Count(Moves) – Count(OpponentMoves)” as its evaluation function. My best three evaluations functions therefore were

1. **ValueScore (VS) / 74.9%**
2. Value2WeightedScore (V2SW) / 73.6%
3. Value3Score (V3S) / 72.3%

Subsequent runs of the Script yielded different Win Rates but all my evaluation functions always outperformed AB\_Improved. It seems more testing would be needed to eliminate a certain randomness in the results. But due to the limited time available no more tests have been performed.